UNIVERSIDAD COMPLUTENSE DE MADRID
FACULTAD DE INFORMÁTICA
Departamento de Ingeniería del Software e Inteligencia Artificial

TESIS DOCTORAL

Análisis de la distribución de las interacciones en la web

MEMORIA PARA OPTAR AL GRADO DE DOCTOR

PRESENTADA POR

Manuel Jesús Gómez Zotano

Directores

Jorge Jesús Gómez Sanz
Juan Pavón Mestras

Madrid, 2018

Análisis de la distribución de las interacciones en la web

UNIVERSIDAD
COMPLUTENSE
MADRID

TESIS DOCTORAL

Manuel Jesús Gómez Zotano

Dirigida por los Doctores
Jorge Jesús Gómez Sanz
Juan Pavón Mestras

Departamento de Ingeniería del Software
e Inteligencia Artificial
Facultad de Informática

Marzo 2017

Análisis de la distribución de las interacciones en la web

Memoria que presenta para optar al título de Doctor en Ingeniería Informática

Departamento de Ingeniería del Software
e Inteligencia Artificial
Facultad de Informática
Universidad Complutense de Madrid

Marzo 2017

A las personas que me han empujado a hacer esta tesis realidad: Beatriz y Margarita.

Agradecimientos

El aburrimiento se cura con curiosidad.
La curiosidad no se cura con nada.

Dorothy Parker

Realizar una investigación y escribir una tesis doctoral no es algo sencillo, y es aún más difícil cuando uno no se dedica a la investigación de forma profesional. Hace falta mucha moral para poder llevar a cabo este trabajo. Desde luego, éste no sería posible sin la ayuda y el empuje de la gente que te rodea y de la gente que te guía.

Es por ello que, en primer lugar, mi mayor agradecimiento no puede ir a otros que a mis dos directores de tesis: Juan Pavón y Jorge Gómez.

Aún recuerdo cómo en unos días de vacaciones, en la Semana Santa de 2012, contacté con Juan y le conté mis circunstancias personales, laborales y mis ganas de investigar. El tema lo quería circunscribir a la web, ya que trabajo en la Corporación Radio Televisión Española (RTVE), concretamente en la Dirección de Tecnología de Interactivos, unidad responsable de la web y sus circunstancias.

Lo que me motivaba del estudio de la web es todo lo que, aún trabajando en dicho dominio, no conocía: el rol de los usuarios, cómo caracterizar su comportamiento, qué impulsa a los usuarios a consumir o cómo ha afectado las redes sociales y los modelos de marketing actuales a un sitio web como rtve.es con tanto contenido y de tan variado tipo.

Juan, a pesar de que no era su área, tomó el reto y me presentó a Jorge, y gracias a ellos, a su constancia, su ejemplo, su sagacidad, su buen criterio y su crítica constante, pero constructiva, me permitió conocer el dominio donde trabajo y en definitiva mejorar como investigador, pero también como analista de datos. Por ello, y por empujarme especialmente en los malos momentos, que han sido muchos, mi más sincero agradecimiento.

Otro punto importante de apoyo para esta tesis ha sido mi propia familia y las personas que me rodean. Han sido cuatro años exigentes en lo personal

y en lo profesional, y gracias al apoyo de mis más íntimos he podido abordar la tarea de hacer esta tesis doctoral.

No me puedo olvidar de quienes de forma desinteresada compartieron conmigo logs e información de distintos sitios web. En concreto mi agradecimiento sincero a:

- Andrés Pedrera Carvajal: Chief Technology Officer en www.bodaclick.com.
- Ezequiel García Collantes: Chief Technology Officer en www.rtve.es.
- Eloy Acosta Toscano: Subdirector de Sistemas Interactivos en www.rtve.es.
- María Maicas Royo: Subdirectora de Aplicaciones y Multi pantalla en www.rtve.es.
- Héctor Escolar: Chief Information Officer en www.bitban.com.
- Javier Polo Gómez: Jefe de Operaciones deportivas en la Unión Europea de Radiodifusión (UER).
- Manuel Mañas y José M. Hernández de Miguel de www.ucm.es.
- Al departamente de IT staff de la Facultad de Informática de la UCM, liderados por Rafael Ruiz Gallego-Largo, y el decano de la Facultad, Dr. Daniel Mozos, por autorizar el uso de los logs.
- Alberto Gómez Chacón: Webmaster & web designer.
- Jerónimo López: Fundador & Chief Tecnhology Officer en www.otogami.com.

Finalmente, gracias también a los proyectos ColosAAL ((TIN2014-57028-R) y MOSI-AGIL-CM (S2013/ICE-3019) por aportar conocimientos y recursos necesarios para el análisis estadístico de los resultados de esta tesis.

Índice

Índice de figuras

Índice de Tablas

Resumen

El auge de Internet, el uso masivo de dispositivos móviles y los cambios en hábitos de consumo de los servicios en la web, han dado lugar a variaciones en los patrones de uso de los sistemas de información respecto de los encontrados en la bibliografía. Uno de los patrones frecuentemente citados corresponde a una función de distribución llamada Zipf (Zipf, 1949). Zipf se ha aplicado a diferentes ámbitos y de forma especial en la linguística. En el contexto de la web, modela cómo las páginas y los recursos de un sitio son solicitados por los usuarios. Dicha distribución establece que si ordenamos los recursos por su frecuencia de acceso, la frecuencia de un recurso es proporcional a su posición en dicha ordenación.

Esta memoria de tesis gira alrededor de este modelo Zipf y motiva cuestiones al respecto. En particular, si sigue apareciendo en el dominio de la web, cuáles son las consecuencias de que exista y cuestionar acerca de los factores que afectan o condicionan dicho comportamiento. Por ello se ha dividido en tres partes: el estudio de la Zipf en el servidor, el estudio de la Zipf en el cliente y la relación entre el comportamiento entre cliente y servidor.

Para determinar el comportamiento en la capa de servidor, se han seleccionado hasta 16 trazas de sitios variados (por tipo de contenido, sector, tráfico, etc.). Tras su análisis, el resultado ha sido que la distribución Zipf aparece en todos ellos, pero que ha habido un cambio respecto a lo que se había dado en la bibliografía: la distribución tiene más sesgo, esto es, en las trazas analizadas hacen falta menos objetos para obtener el mismo porcentaje acumulado de frecuencias que en las trazas anteriores. Esto lleva a que en los sitios web actuales, la popularidad de los objetos está distribuida de una forma más abrupta, con algunos objetos presentando una frecuencia altísima, y una inmensa mayoría de ellos con frecuencias muy bajas. Esto ocurre con independencia del sector, de la tecnología o del propósito del sitio.

El estudio de la distribución Zipf en el servidor se ha complementado con un análisis más exhaustivo del comportamiento de uno de los sitios seleccionados (www.rtve.es), en el cuál se ha podido ver que la división de un log por horas o por tipos de contenidos da como resultado distribuciones basadas en

la Zipf. Además se muestra que para dicho sitio también se pueden encontrar otras distribuciones Zipf, como por ejemplo al analizar el número de robots que leen sus páginas, o la distribución de peticiones que realiza cada robot.

Una de las consecuencias que se muestran en la tesis es el hecho de que para sitios que siguen una Zipf se pueden construir caches súper eficientes. Para ello basta utilizar la popularidad como factor, pudiéndose alcanzar un hit ratio muy alto con un volumen muy reducido de recursos.

Otra de las consecuencias que se analiza en esta tesis del hecho de que las distribuciones sean tan agudas, es que permite acotar el máximo teórico de una configuración de cachés, lo que implica que en se puede predecir, en base al comportamiento que se detecta en el servidor, cuál será la eficiencia máxima de cachear.

Hay una relación entre lo que ocurre en el cliente (esto es en los navegadores o en aplicaciones móviles) y lo que se aprecia en el servidor. Una página que se pide por un navegador provoca la petición de recursos adicionales tales como imágenes, hojas de estilos, etc. Por ello se ha analizado la capa de cliente de un sitio, RTVE. Se ha encontrado que las peticiones que se hacen desde cliente a páginas web siguen distribuciones basadas en Zipf. Dichas distribuciones muestran un sesgo enorme entre las páginas, y como en el caso del servidor, hay objetos que son muy demandados frente a otros que apenas lo son.

Este resultado verifica que en el sitio bajo estudio se da el comportamiento del *long tail*, donde la inmensa mayoría de las páginas son pedidas de forma anecdótica, y una porción muy pequeña de contenidos concentra el grueso de la atención de los usuarios del sitio. El sesgo encontrado de los contenidos se puede explicar por el cambio de modelo de consumo de los usuarios provocado por cómo se consume en las redes sociales, por los buscadores y por el modo de consumir en móviles.

Finalmente, la tesis ha abordado la relación entre lo que se puede ver en el cliente y cómo se manifiesta en el servidor, en términos de qué tipo de función de distribución se da en el servidor. La hipótesis es que en www.rtve.es se ha encontrado una Zipf en el servidor, porque las páginas en el cliente siguen una Zipf. Para demostrar si es cierto se ha hecho una simulación con www.rtve.es, donde se utilizan como entrada conjuntos de páginas web que siguen distribuciones aleatorias, Zipf, uniformes y normales. El resultado es que en todos los casos se da una Zipf con independencia de cuál sea el comportamiento del usuario. Esto nos lleva a concluir que un sitio web transforma una distribución de páginas en cliente, en una distribución de recursos web que siguen una Zipf en el servidor, y por tanto, el comportamiento del usuario únicamente varía la pendiente de la Zipf del servidor.

El análisis anterior se ha hecho también con el sitio web www.wikipedia.org, un sitio con una tecnología y un propósito diferente a www.rtve.es. Sin embargo, a pesar de esta diferencia, el resultado ha sido el mismo. De nuevo, con independencia del comportamiento del usuario en el cliente, en el servidor siempre se de una Zipf.

La conclusión a la que se llega es que el comportamiento de Zipf en el servidor se debe esencialmente a elementos estructurales del protocolo HTTP y de cómo se conciben las páginas HTML.

Summary

The rise of the Internet, the massive use of mobile devices, as well as changes in the consumption habits in the web, have led to variations in the usage patterns of information systems with respect to the literature. One of those patterns has to do with the Zipf distribution(Zipf, 1949), which has been applied to many domains, and remarkably in linguistics. In the context of the world wide web, it models how web pages are requested by users. This distribution states that if the resources of a website are sorted by the number of received requests, then the frequency of each resource is proportional to its position in that ranking.

This thesis addresses this Zipf model in the world wide web and raises some questions. Concretely, if Zipf still models received requests, as well as the consequences that this entails, and, should this still holds, what factors are affecting or conditioning such behavior. To this end, the thesis has been divided into three parts: the analysis of Zipf distribution's presence in the server side, the analysis of Zipf distribution's presence in the client side and, finally, the relation between the behavior in client and its consequences in the server.

To determine the server-side behavior, 16 logs from diverse websites have been collected. After their analysis, a Zipf distribution has appeared in each of them. Compared to the results from previous studies, the requests distributions in this thesis show a more skewed probability distribution than in the past. Therefore, there is a higher concentration of requests in fewer objects and a higher volume of resources with a very small number of requests. This pattern happens regardless of the technology, purpose or sector of the website analyzed.

In the server side, a deeper analysis has been made for one of the websites under study (www.rtve.es). It shows that the subsets, that are obtained by splitting the log either by hour or content type, follow a Zipf-like behavior, where each subset has its own degree of skewness and objects' concentration. Additionally, other Zipf-like behaviors have been found on it: for instance, the analysis of the requests made from every robot to www.rtve.es, or the

frequency of pages that are requested by robots, show a Zipf distribution for both cases. As it can be seen, Zipf appears in many different situations in the web domain.

The first consequence of these results is that super efficient caches can be implemented using a few resources for websites with such degree of skewness. The hit ratio that can be obtained is close to a 90 % using only a 0.6 % of the working set (size occupying all the objects to distribute).

The second consequence is the predictability feature: the maximum theoretical hit ratio can be calculated based on the current behavior of the website, regardless of the skewness of the requests distribution.

The thesis assumed as working hypothesis that there should be a relation between the client behavior (that is, the web pages requested in a browser or from a mobile application) and the Zipf behavior detected in server. The RTVE website has been studied to find correlations between these two layers. The experiments on the client side show a Zipf behavior with a high skewness, similar to the one shown on the server side. The reason maybe that web pages are mostly requested individually, following, perhaps, references obtained from social networks, from search engines, and from how the pages are consumed in mobiles.

After the previous result, the relation between layers have been revised and the working hypothesis questioned: our hypothesis is that the cause of having a Zipf in server logs is that client requests to web pages follow a Zipf too. In other words, a Zipf is found in the server because there is a Zipf in the requests from client. A simulation has been built to verify the hypothesis: webpages from the RTVE website have been requested using different client distributions, based on Zipf, random, normal and uniform distributions. The result is that no matter the web pages distribution in the client, a Zipf is always found in the server side. It means that a website transforms the clients web pages distribution into an object distribution following a Zipf pattern in the server, where the users' behavior only defines the degree of skewness of the distribution.

The simulation was repeated using the www.wikipedia.org website, which returned similar behavior. Our conclusion is that Zipf-like behavior on the server side is due to the way websites work based on the HTTP protocol, and how the pages are built based on the concepts defined in Hypertext Markup Language (HTML). User behavior in itself does not explain the Zipf behavior seen in the server.

Glosario

AWS Amazon Web Services. 1, 85

CDN Content Delivery Network. 2, 3, 73, 85

CLF Common Log Format. 25, 26, 43, 53, 85

CSS Cascading Style Sheet. 49, 85

GD Greedy Dual. 18, 85

GDS Greedy Dual Size. 12, 14, 42, 47, 85

GDSF Greedy Dual Size Frequency. 12, 14, 15, 42, 43, 45, 47, 85

HTML Hypertext Markup Language. 3, 6, 9, 21, 23, 67, 68, 73, 85

HTTP Hypertext Transfer Protocol. 6, 8–10, 24, 26, 31, 54, 85

IP Internet Protocol. 25, 30, 31, 85

LFU Least Frequently Used. 12, 14, 15, 18, 19, 42, 45, 47, 85

LRU Least Recently Used. 11, 14, 15, 42, 47, 85

MB Mega Byte. 10, 42, 43, 45, 47, 85

OJD Oficina de Justificación de la Difusión. 50, 85

PC Ordenador Personal. 1, 26, 85

RTVE Corporación Radio Televisión Española. 3, 33, 43, 49–53, 55, 59, 70, 72, 75, 85

SSL Secure Socket Layer. 9, 85

URL Universal Resource Locator. 6, 9, 26–29, 31, 49, 53, 54, 56, 61, 62, 85

Capítulo 1

Introducción

1.1. Motivación

La diversificación en los medios de acceso a la información, así como la personalización de los servicios, ha cambiado la forma de ver y utilizar Internet. Se ha pasado de un World Wide Web accedido únicamente vía el navegador a un ecosistema de dispositivos diversos que acceden a unos servicios de información comunes. Asimismo se ha pasado de un modelo de sitios con información estática, a una web donde se proveen servicios personalizados. Estos cambios pueden suponer una alteración del hábito de consumo, ya que el usuario tiene la posibilidad de acceder directamente a la información y a los servicios que requiere, cuando antes se accedía a sitios web que agregaban contenido y donde el contenido era igual para casi todos los usuarios.

El aumento del número de usuarios conectados, así como el ofrecer servicios cada vez más personalizados, ha supuesto un incremento en el número de peticiones que los sistemas de información, como los sitios web, tienen que resolver. Esto se ha traducido en una mayor preocupación por la escalabilidad de los sistemas de información para poder hacer frente al incremento de usuarios y de medios que estos emplean para acceder a la información.

En la actualidad hay dos formas de enfocar dicha escalabilidad en sistemas de información(Podlipnig y Böszörmenyi, 2003): distribuyendo la carga, utilizando por ejemplo servicios distribuidos como Amazon Web Services (AWS), o bien distribuyendo los contenidos en la forma de ficheros utilizando caches distribuidas, tal y como hacen las Content Delivery Network (CDN). El mecanismo de escalabilidad se basa en la distribución bien de la carga, bien del contenido, pero, en cualquier caso, siempre aumentando el número de equipos que intervienen en una escala global.

De los dos sistemas, este trabajo de tesis se centra en las cachés distri-

buidas. Este tipo de soluciones tienen sentido en los casos donde un mismo contenido es consumido por diferentes usuarios. En este sentido, son paradigmáticos los sitios web de noticias y multimedia, que tradicionalmente emplean un enfoque basado en CDN. Estos servidores ofrecen el mismo contenido a diversos usuarios y gestionan especialmente bien aquellos que son más pesados, como los vídeos. Así los vídeos se replican entre múltiples servidores, con lo que disminuye la latencia en el servicio y aumenta la escalabilidad. Atender más usuarios con menos recursos es un elemento crítico en la estrategia empresarial de la industria de contenidos. Es también algo necesario dentro de las últimas tendencias en sostenibilidad de los recursos informáticos. En esta filosofía, que persiguen hacer un mejor uso de instalaciones y reducir el consumo energético (Hooper, 2008).

La hipótesis de trabajo de esta tesis es que estos sistemas de cachés serían más eficiente en consumo de recursos y en coste económico si se tuviera un mejor conocimiento de lo que solicitan los usuarios. Si el objetivo de los sistemas de información es proveer contenido a los usuarios, tiene sentido el estudio de cómo los usuarios se comportan ante tales sistemas en términos de qué solicitan y con qué frecuencia.

El interés de analizar el comportamiento de los usuarios ante sistemas de información tiene precedentes en la literatura (Breslau et al., 1999; Krashakov et al., 2006). Estos análisis previos del comportamiento parten de colecciones de datos obtenidos o bien mediante servidores intermedios (*proxies*) o mediante el análisis de los registros de peticiones (*logs*) de los servidores bajo inspección. En la literatura (Breslau et al., 1999; Krashakov et al., 2006; Roadknight et al., 2000; Shi et al., 2005) hay un consenso sobre que la ley que gobierna el modelo de consumo y la forma de distribución de los contenidos en los sitios web en el lado de servidor es la ley exponencial inversa (ver sección 2.3) y más concretamente basados en una Zipf (Zipf, 1949) que se caracteriza por un parámetro *alpha*. Zipf (Zipf, 1949) se ha aplicado a diferentes dominios, siendo la linguística uno de los más citados. Dicho modelo aplicado a este contexto de modelado de peticiones de usuario en sistemas de información establece que la frecuencia de acceso de un contenido decrece de forma exponencial inversa. Así, la mayor parte de los accesos de un sitio se concentra en un conjunto pequeño de contenidos, mientras que la mayoría de los objetos de un servicio web son poco accedidos. Desde un punto de vista industrial y desde el servidor esto implica que la popularidad es un factor crucial a la hora de establecer políticas de cache (Breslau et al., 1999).

A pesar de la corrección de tales estudios, caben dudas sobre la vigencia de los mismos y de si sus conclusiones son válidas a día de hoy por dos motivos. Primero, porque la mayoría de los estudios se realizan en instituciones

académicas y en ambientes muy controlados. Segundo porque, comparativamente hablando, tanto el tráfico, es decir, el número de peticiones a observar, como el número de usuarios y el volumen de recursos y páginas solicitadas son mucho mayores en la actualidad que en los análisis de la literatura, con lo que también lo son los *logs* que se obtienen. Aparte, los contenidos ya no se acceden exclusivamente desde navegadores. El propósito de los sitios de información se ha generalizado a distintos dispositivos y la penetración de la web en la actualidad también es significativamente mayor, dando la posibilidad de introducir en los estudios variables como el sector de consumo asociado del sitio web o su tecnología. En este contexto, parece posible el que se puedan encontrar comportamientos distintos.

Estas dudas se traducen en tres cuestiones que se desarrollan en esta tesis. Primero si todavía se puede asumir que el comportamiento de un usuario se puede describir con una Zipf atendiendo a los logs existentes hoy en día y usando una colección de datos de mayor tamaño y variedad. Segundo, si esta distribución afecta exclusivamente a los objetos web que componen una página, o bien si son extensibles también a los URL únicos que representan la página visitada (las páginas web). Tradicionalmente, el modelado Zipf se ha estudiado sólo sobre los recursos u objetos web que son accedidos en un entorno web sin considerar de forma aislada las páginas web, que son, en definitiva, los elementos sobre los que los usuarios de un sitio hacen una elección y deciden consumir. Tercero, en caso de ser válida una distribución Zipf en ambos casos, si el valor obtenido experimentalente de α en el pasado sigue manteniéndose como en la literatura. La importancia de este dato radica en que dicho valor condiciona la política de gestión de cachés. Sobre esto último, en principio, el hecho de tener colecciones con mayor volumen de peticiones posibilita una mayor estabilidad de α, tal y como se recoge en Breslau et al. (1999). No obstante, no se han encontrado trabajos donde se analice en profundidad y desde diferentes perspectivas este hecho.

El alcance e importancia de estas cuestiones está relacionado con una mejor configuración de las estrategias y recursos disponibles para sistemas de caché. Se mostrará que son problemas que siguen abiertos y que una mejor comprensión de los mismos conllevará una mayor eficacia y capacidad de llegar a más personas. En este sentido, el análisis a realizar será exclusivamente estadístico, sin atender a variables socio-demográficas ni contener sesgos socio-culturales.

El único sesgo que se reconoce en este trabajo es que el uso de Zipf implica que hay un conjunto de objetos que son altamente demandados y accedidos. Esto quiere decir que el interés de los consumidores se centra en una muestra pequeña de objetos, mientras que la mayoría de los contenidos son de acceso

residual. Este comportamiento de consumo va en línea con lo que ocurre con el contenido que se comparte en las redes sociales, donde hay contenidos que tienen un éxito rotundo mientras que la inmensa mayoría de los demás pasan inadvertidos. Por ello, cobra interés el análisis del lado de cliente, para ver si el consumo de cliente en los sitios se ha vuelto más exigente o compulsivo.

1.2. Objetivos

En esta tesis se parte de la hipótesis de que los sistemas de cachés serían más eficientes en consumo de recursos y en coste económico si se tuviera un mejor conocimiento de lo que solicitan los usuarios. Aunque existen resultados previos que avalan la existencia de una distribución Zipf como representativa de este conocimiento, se argumenta que el conocimiento es mejorable si se logra determinar variaciones modernas del parámetro *alpha*.

Así pues, la novedad de los resultados se fundamenta en el acceso a fuentes de datos de importancia no disponibles antes en la literatura. Es el caso de los logs de RTVE y de otros grandes medios, como diarios digitales de gran tirada. Contrastar los resultados de RTVE con otros servicios web en varios ámbitos es relevante. Para tal fin se plantea analizar sitios similares, esto es, sitios de contenido multimedia del sector audiovisual, pero también hacer extensiva la comparación con servicios que no tienen nada en común. Además de lo anterior también se plantea validar los resultados con servicios de otros sectores y que empleen otros tipos de tecnologías. El hecho de contar con una muestra tan heterogénea de sitios web permite obtener algunas conclusiones y resultados de carácter general, así como dotarnos de un conjunto de *logs* para hacer análisis comparativos, no sólo desde el punto de vista del comportamiento del usuario, sino también para estudios relativos a cómo se modelan los sitios o cómo se destacan unos contenidos respecto de otros.

Siguiendo las cuestiones de la introducción, se plantean los siguientes objetivos de investigación, dentro de los cuales se desglosan varios objetivos secundarios:

- Determinar si las distribuciones tipo Zipf siguen modelando las frecuencias del acceso a objetos web en el lado de servidor con los volúmenes actuales de información y las tecnologías vigentes. En caso afirmativo, el trabajo aborda cinco aspectos adicionales:
  - Si se dan distribuciones tipo Zipf con independencia del sector del sitio web o de la tecnología subyacente.
  - Si el valor α (ver sección 2.3) está acotado entre 0.6 y 1(Breslau et

al., 1999), como indica la literatura, o si su valor se ha modificado de forma significativa.

- Si hay valores nuevos de α, si estos modifican o condicionan los resultados de la literatura en lo que se refiere a cuál es la mejor estrategia de caché.
- Si la descomposición Zipf en servidor por horas o por tipo de contenido, da lugar a otras distribuciones que también tiene un comportamiento basado en Zipf.
- Evaluar cuál es el impacto de clientes tipo robot en la distribución de un sitio web.

- Determinar si las peticiones del cliente, esto es las páginas web más que los objetos web, también siguen una Zipf.
  - En caso afirmativo, determinar si las estructura de las peticiones del clientes son las responsables de la existencia de una Zipf en el lado del servidor, o bien es algo que puede suceder a pesar de que un cliente no realice tales peticiones según esa distribución. Por ejemplo, se si un cliente realiza peticiones cuya frecuencia sigue una distribución normal o uniforme, se pregunta si, pese a ello, en el servidor se seguirá observando una distribución Zipf de los objetos web.
  - Si existe alguna correlación entre lo que ocurre en el lado del cliente y en el lado de servidor. En particular, se estudian:
    - si el *alpha* observada en la distribución Zipf de lado del cliente está relacionada con la observada en el lado del servidor. No es una pregunta trivial pues se desconoce a priori cuántas peticiones de objetos web se crean desde una página web.
    - si existe dicha correlación qué lo justifica.

1.3. Metodología de trabajo

Este trabajo de tesis realiza varios estudios empíricos para responder a las cuestiones planteadas. Estos estudios se realizan atendiendo a la capa de cliente, también citada a veces como capa de vista, y a la capa de servidor. En ambos casos, se buscará analizar el comportamiento del usuario recopilando datos de su actuación y comparando los resultados contra los obtenidos en la literatura.

En la capa de cliente o de vista, el análisis del comportamiento del usuario se realiza mediante el muestreo de las acciones y de la navegación que éste realiza (Calzarossa et al., 2016). Para ello se han desarrollado múltiples herramientas o servicios (ver sección 2.2.2) que permiten hacer el seguimiento del usuario. Los puntos más relevantes son la identificación del usuario único, la sesión y la página vista (Cloud, 2016; Mahanti et al., 2009; Calzarossa et al., 2016). En la bibliografía no se ha encontrado ningún trabajo exhaustivo analizando y caracterizando cómo se comporta el usuario en esta capa. El motivo principal es la dificultad en el acceso a dicho comportamiento en sitios webs altamente demandados, ya que esta información es extremadamente sensible y afecta al modelo de negocio.

Sobre la capa del servidor, lo registrado es consecuencia de acciones desde la capa de cliente. En la literatura hay artículos que tratan de analizar el comportamiento en la capa de servidor, utilizando diversas técnicas, descritas en el capítulo 2.2.1. Los resultados que se obtienen al analizar esta capa tienen un interés industrial ya que permiten planear la mejor estrategia técnica a la hora de montar el servidor que de soporte. Desde el punto de vista científico permite validar y refutar las métricas y modelados encontrados en la literatura, así como los empleados en los bancos de carga (workloads) usados en simulación.

El análisis de ambos aspectos es similar, variando especialmente la extracción de datos. El primer paso que se ha hecho en esta tesis es realizar un estudio del estado del arte para analizar trabajos relacionados e identificar los medios de trabajo que aceptan los antecedentes. De entre ellos, destaca la recolección de colecciones de datos para su posterior análisis, llegándose a la conclusión de que dichos bancos son muy antiguos (Krashakov et al., 2006). Por tanto, vista la ausencia de bancos de datos actuales de servidor, se ha procedido a la recolección de los mismos contactándose directamente con los administradores de sitios, quienes han provisto de la información en forma de logs.

En la parte de servidor se suelen emplear los datos que se registran en el servidores en la forma de logs. Dichos logs contienen todas las peticiones que se hacen el servidor y se guardan con un formato estandarizado, el formato *Apache* (ver sección 2.2.1). En sistemas que emplean más de un nodo para servir el contenido, ha sido necesario hacer la fusión de los logs utilizando la fecha y hora de cada petición para ordenar las peticiones cronológicamente.

Posteriormente, estos datos se han utilizado para hacer simulaciones y determinar la conveniencia de distintas políticas de caché en los servicios analizados, poniendo el foco en aquellos sitios con un mayor volumen de peticiones. De esta forma se ha podido validar el conocimiento previo respecto

del caché en los sitios bajo análisis.

El siguiente paso ha sido analizar el estado del arte en la capa de vista, y viendo si hay o no colecciones de datos. La conclusión es que en esta capa se utilizan herramientas más sofisticadas como Google Analytics o Adobe Omniture, las cuales registran las peticiones de los usuarios. Se ha procedido a recabar datos para varios dominios, y su correspondiente análisis (ver sección 2.2.2). Para poder hacer una correlación entre cliente y servidor, se han buscado juegos de datos que sean de las mismas fechas.

Finalmente, tras los análisis, con los resultados se ha procedido a parametrizar y caracterizar los comportamientos tanto en cliente como en servidor, para en último paso, correlacionar comportamientos entre capas y extraer hechos que expliquen los comportamientos y correlaciones encontrados.

El análisis estadístico de la información en todos los casos se ha hecho con R. Se ha perseguido corrección a la hora de evaluar cada aspecto, especialmente los de correlación.

1.4. Organización

El capítulo 2 se dedica al estado del arte. En este estado del arte se estudia la importancia de las caches en los sistemas de información actuales y se ilustra la importancia de las políticas de gestión. Estas políticas son dependientes del comportamiento esperado del usuario a la hora de reclamar contenidos. A continuación se justifica la singularidad de Zipf como ejemplo de powerlaw que sirve para modelar el comportamiento de estos usuarios. Se revisa la literatura al respecto para demostrar que: (1) aunque no hay una parametrización uniforme de α, existe una rango de valores que definen dicho α alrededor de 1 y que se han usado para entender el tráfico web hasta ahora; (2) que el hecho de ser una Zipf influencia en la efectividad del tipo de estrategia más utilizada por la cache. Caben varias preguntas de investigación sobre los resultados de Zipf respecto a la influencia del metodo de obtención de datos, de la perspectiva del cliente o del servidor o del impacto de la tecnología en el valor de α.

El capítulo 3 realiza un estudio empírico actualizado de la presencia de la función Zipf en las trazas de servidores de gran tráfico discutiendo el impacto de las tecnologías. El método utilizado para analizar la traza permite determinar una nueva parametrización actualizada del parámetro α que resulta ser aproximadamente un 50 % más de lo indicado en la literatura previa. Se realiza también un análisis de la evolución del parámetro α de Zipf a lo largo de las franjas horarias. Este análisis indica que hay una desviación estándar

de 0,07 sobre el alfa encontrado. Parte del estudio incluye el descartar el efecto que puedan tener bots y arañas automáticos en estos resultados.

El capítulo 4 se centra en el análisis del impacto del parámetro α en la configuración de políticas de gestión de caches. Estos resultados se contrastan con destacados en la bibliografía con el fin de validar resultados actuales y revisar para valores de α superiores a 1 si hay conclusiones diferentes de las observadas. Esto último tiene sentido toda vez que la experiencia registrada era que α era aproximadamente 1. Demuestra, mediante una serie de simulaciones basadas en logs reales, qué comportamiento habría que esperar en función de α, qué política puede ser más apropiada.

El rendimiento en algunos casos es inesperado, como con RTVE que consigue gestionar el 90 % de las demandas con 21MB de cache. Siendo un sitio web de varios millones de accesos. Este resultado supera con creces a los encontrados en la bibliografía, y se ve explicado por el desarrollo matemático que lo predice, también desarrollado en el capítulo 4.

El capítulo 5 trata el punto de vista del cliente. Los capítulos 3 y 4 han analizado los logs de los servidores atendiendo a todos los elementos de una página (como texto, vídeos, imágenes o guiones). En este capítulo se adopta un enfoque distinto, revisando URLs que engloban todos los elementos anteriores. Atendiendo a estos URLs, y analizándolos con el mismo método aplicado en el capítulo 3, se determina que también estas URLs, y por tanto el comportamiento de los usuarios a la hora de seleccionar las páginas a consumir, siguen una Zipf. Este resultado es original, ya que no hay publicaciones que analicen las páginas web de los sites debido a que son datos que ningún webmaster publica.

La magnitud del *alpha* observado supera también el valor *alpha* identificado en el estado del arte. Esto confirma nuevamente las conclusiones del capítulo 3.

El capítulo 6 se centra en la generación de peticiones desde el lado del cliente. Se plantea si una reordenación de las peticiones en forma de URLs siguiendo otro tipo de distribuciones, como una normal, una uniforme o una Zipf con un *alpha* menor de 1, generan en el lado del servidor o no una Zipf con valores similares a los obtenidos experimentalmente. Los experimentos realizados indican que, con independencia de cuál sea la distribución de las URLs en el lado del cliente, siempre aparece una Zipf en servidor. La conclusión es que hay indicios de que la Zipf no depende de cómo se hacen las peticiones desde un cliente, sino de la estructura de la página o del modo en qué se relacionan los objetos web dentre de las páginas web. Este resultado es novedoso, ya que no se ha encontrado ningún estudio ni resultado similar siguiendo el enfoque aquí descrito.

La tesis concluye con el capítulo 7 con las contribuciones principales y cómo se mejora el estado del arte con el trabajo realizado.

Capítulo 2

Estado del arte

En la introducción se han identificado como objetivos prioritarios el analizar la adecuación de la función Zipf para representar accesos a objetos web desde el lado del servidor y también el análisis desde el lado del cliente. En este estado del arte se aborda la literatura relacionada con ambos aspectos y justifica la necesidad de una revisión de estos resultados.

En el análisis de estas funciones Zipf es importante la motivación original, que está relacionada con las infraestructuras de sistemas de información, y más concretamente, en las cachés dentro de redes de distribución de contenidos. La relación entre cachés y Zipf ya ha sido señalada en la literatura (Almeida et al., 1996; Breslau et al., 1999). No obstante, dado que parte de experimentación de esta tesis consiste en reproducir políticas de reemplazo en cachés bajo diferentes condiciones, es necesario explicar cómo funcionan las cachés y qué resultados arroja la literatura especializada en este ámbito. Los bancos de prueba para rendimiento de cachés son habituales. Dado que la tesis se centra en la relación entre Zipf y políticas de reemplazo de cachés, la revisión del estado del arte en este punto se centrará en estudios sobre la relación entre el espacio de trabajo y el tamaño de la caché versus el rendimiento de las políticas. Además, como se señalará en este trabajo cuando se revisen las funciones Zipf, los bancos de peticiones que se han usado en los estudios publicados sobre comportamiento de cachés atendiendo a la función Zipf son antiguos (Krashakov et al., 2006). Ello justificará la necesidad de validar las políticas de caché más habituales utilizando colecciones de datos modernas.

La distinción del objeto de estudio, bien sea objetos web o bien páginas web, hace necesario aclarar qué es cada uno en el ámbito de esta tesis. La frecuencia de aparición de uno u otro afecta a la forma de la función Zipf cuya existencia la literatura ha mostrado y que este trabajo de tesis quiere revisar

debido a los cambios tecnológicos y sociales. Y esta frecuencia afecta a su vez a la configuración de los servicios de información que deben proveerlos. Como se verá en la propiedad 1, la forma de la función Zipf es relevante a la hora de configurar un sistema de cachés. Hay que señalar que las cachés almacenan objetos web, no páginas web. Un cliente consulta páginas web, pero solicita al servidor objetos web. Esta distinción obliga a separar los dos aspectos identificados en los objetivos de la tesis y a su estudio separado.

Definición 1. *Una **página Web** es una página HTML accesible por los usuarios a través de una Universal Resource Locator (URL) utilizando el protocolo HTTP. Una página Web consiste en una noticia, la página de un vídeo, una portada agregadora de contenidos, etc. En aplicaciones nativas, las páginas también se refieren a la vista nativa dentro de una aplicación del contenido correspondiente identificado por la misma URL. De esta forma se puede contabilizar el consumo de un contenido con independencia de la vista.*

Definición 2. *Un objeto web es todo elemento que es accesible mediante una URL. Incluye ficheros CSS, imágenes, vídeos, páginas Web, Javascripts, objetos Flash, etc.*

La discusión sobre la función Zipf se entiende mejor si viene precedida de un contexto tecnológico, los problemas que existen en dicho ámbito, así como los métodos de trabajo seguidos en ese área. Por ello, el estado del arte comienza con una discusión de resultados relevantes en el área de arquitecturas y políticas de gestión de cachés para objetos web. Se explicarán qué son la cachés, y dentro de esta área, los elementos más habituales para su análisis: métricas, factores, definición de cacheable en el dominio de la web, políticas de reemplazo y finalmente algunos resultados existentes en la bibliografía donde se compara el rendimiento de distintas políticas de cache.

A continuación se va a definir qué es una Zipf, qué implica serlo y las evidencias encontradas de su existencia tanto a nivel de objetos web como de páginas web. Como parte del estado del arte hay que justificar la importancia de aclarar cuál es el rango en que se mueve esta función a día de hoy. En este análisis preliminar, se aclara qué preguntas quedan sin responder respecto a la causa de que exista esta función y qué experimentos podrían ser pertinentes.

El trabajo de tesis requiere también explicar qué técnicas se pueden emplear para analizar la frecuencia de aparición de peticiones de objetos web o de páginas web. Por ello, se dedica una sección a describir técnicas habituales en este tipo de trabajos.

2.1. Cachés en el domino de la Web

El empleo de cachés en servicios Web es una técnica muy extendida en la actualidad ya que permite mejorar la calidad de los servicios al minimizar la carga de los servidores finales (Podlipnig y Böszörmenyi, 2003). Esta técnica está basada en el hecho de que cada petición se refiere a un objeto web accesible a través del protocolo HTTP, de tal forma que dicho objeto puede ser utilizado para dar respuesta a otros usuarios. La técnica se basa en que cuando una petición llega a un servicio, el objeto web se resuelve por el servicio, y una copia del mismo es almacenada para futuras peticiones. Cuando una nueva petición llega al servicio Web, el objeto es buscado en la caché, y si es encontrado se sirve desde ésta, evitando tener que acceder al servidor final para obtener el objeto pedido.

El uso de cachés en la web, así como su impacto en cliente y servidor, ha sido ampliamente estudiado en el dominio de la Web. Los estudios han identificado diferentes aspectos que tienen que ser considerados para la correcta configuración de una caché. Los más relevantes son el número de objetos a cachear, la distribución de las peticiones a objetos web que los servidores reciben, el tamaño de la caché, la posición en la arquitectura y las políticas de reemplazo a utilizar.

Cuando se valora una caché para un conjunto de datos, hay dos perspectivas que se tienen que tener en cuenta: una estática, es decir, qué objetos se van a pedir, cuántas veces se piden (su frecuencia) y cuáles son los más pedidos; la otra dinámica o temporal, que hace referencia a cuándo se las peticiones a los objetos se van realizando. Esta segunda perspectiva suele ser llamada tráfico o carga de datos.

Esta sección versará sobre cómo los diferentes aspectos que intervienen en una caché han sido analizados en la bibliografía. En el primer punto se define qué peticiones son cacheables dentro del dominio de la Web, esto es, qué condiciones tiene que cumplir una petición a un objeto Web como para que se pueda cachear. A continuación, se presenta el concepto de tipo de contenido HTTP, el cuál se emplea en estudios para determinar propiedades diferenciadas entre tipos de objetos web. Luego, se analizarán las políticas de reemplazo, esto es, el mecanismo que decide cuando un objeto se borra de la cachés, y se ahonda en las características que se utilizan para definirlas.

El siguiente punto trata cómo medir el rendimiento de una caché, esto es, qué métricas se emplean. Entre ellas, destaca el hit ratio (ratio de acierto) como la más utilizada en la literatura.

La última sección muestra algunos trabajos previos que analizan las políticas de caché y cuáles son mejores a partir de evidencias encontradas por

los autores.

2.1.1. Condición de cacheable

En el dominio del Web, las cachés almacenan peticiones de usuarios a objetos web para su uso posterior. En este dominio, no todas las peticiones que llegan a un servicio Web pueden ser cacheadas. Para determinar qué condiciones tiene que cumplir una petición para serlo, es necesario entender qué la caracteriza: hay varios elementos que definen una petición, entre otras, el método y el código de respuesta (Fielding et al., 1999).

El método de una petición indica qué operación el usuario quiere realizar sobre el objeto que solicita. Según Fielding et al. (1999), una petición puede presentar los siguientes métodos:

- **GET**: se emplea para indicar que se quiere recuperar un objeto web. Este método es definido como idempotente, esto es, si dos usuarios cualesquiera realizan una petición al mismo recurso utilizando *GET*, ambos obtendrán idénticas respuestas, salvo si el objeto caducó entre ambas peticiones.
- **POST**: el uso de este método está relacionado con el envío de información desde formularios en páginas HTML.
- **HEAD**: este método es equivalente a *GET* salvo que no devuelve el objeto pedido. Se utiliza para recuperar únicamente meta información asociada a los objetos web.
- **PUT**: se emplea en sistemas que permiten el borrado y modificación de objetos utilizando HTTP. En este caso, una petición que emplee el método *PUT* intenta modificar el objeto indicado en la URL por el contenido presente en el cuerpo de la petición.
- **DELETE**: similar al caso anterior, pero se emplea borrar el objeto indicado en la URL.
- **TRACE**: permite al cliente determinar qué se ha recibido en el servidor.
- **CONNECT**: se reserva para ser usado en proxies y túneles Secure Socket Layer (SSL).

Por su parte, el código de una petición se emplea para comunicar al cliente que la realizó el resultado de la misma, esto es, si ésta fue resuelta satisfactoriamente, si hubo error, si se redirige a otro recurso, etc.

Fielding et al. (1999) y más recientemente Romano y ElAarag (2011) notaron que tan sólo las peticiones Web que presentan un método de petición *GET* tienen que ser cacheadas. De entre éstas, sólo pueden ser cacheadas aquellas cuyo código de respuesta es 200, 203, 206, 300, 301 o 410 . En la tabla 2.1 se muestran estos códigos HTTP junto a su significado.

Tabla 2.1: Códigos de respuesta HTTP

Código	Descripción
200	Documento devuelto correctamente
203	Información no autoritativa
206	Contenido Parcial
300	Múltiples opciones de redirección
301	Movido permanentemente
410	Ido

Algunas de las peticiones que cumplen con las dos condiciones de arriba podrían no ser cacheables, dependiendo de cómo se haya implementado el mecanismo de respuesta en el servidor: las peticiones Web pueden depender de cookies (Fielding et al., 1999) o de atributos de cabecera para generar el contenido. Fielding et al. (1999) notó que el registro de peticiones que aparecen en los logs no incluye la información de cabecera ni las cookies, con lo que debe asumirse la existencia de peticiones no cacheables en los análisis. Esta asunción no es un obstáculo, ya que las trazas, en cualquier caso, pemiten determinar cuál es el valor de hit ratio o tasa de acierto máximo que se puede dar en la muestra (Arlitt y Jin, 2000). De cualquier forma, en las arquitecturas y sitios Web actuales se intenta minimizar los objetos que depende de las cookies, incluso en el caso de servicios logados. Así, el número de peticiones que presentan este problema es pequeño, y el sesgo en los resultados obtenidos es muy limitado.

2.1.2. Tipo de contenido

El tipo de contenido *indica el tipo de formato de datos que es enviado al receptor (por ejemplo, un explorador) o, en el caso del método HEAD, indica el tipo de formato que sería enviado en caso de realizar una petición GET* (Fielding et al., 1999). El protoclo HTTP utiliza el campo de cabecera *content-type* para indicar al cliente el tipo de contenido que el cliente va a recibir, para que éste pueda adaptar adecuadamente los mecanismos de visionado al contenido recibido. Para este propósito se creó una categorización basada en los siguientes 5 tipos de contenidos (Fielding et al., 1999):

- *Text*: indica que el contenido es enviado como contenido textual.
- *Image*: indica que el contenido es una imagen.
- *Audio*: indica que el contenido es un formato de audio.
- *Video*: el cuerpo de la petición contiene imágenes que cambian con el tiempo, y que incluyen color y sonido.
- *Application*: se emplea para indicar que el cuerpo de la petición consiste en datos discretos que deben ser procesados por algún tipo de aplicación.

En general, el tipo de contenido *text* consiste en una secuencia de caracteres que son enviados al cliente, que no proporciona ni permite comandos de formateo, especificaciones de atributos de fuente, instrucciones de procesamiento, directivas de interpretación o marcado de contenido (Freed y Borenstein, 1996).

Para contenido multimedia hay tres tipos de contenidos: *image*, *audio* y *video*. El receptor requerirá de los elementos necesarios para poder visualizar una imagen (en el caso de *image*), para hacer sonar un audio (en caso de *audio*) o la capacidad de hacer que imágenes se muevan y se sincronicen con el audio embebido (en el caso de *video*).

Finalmente, el tipo de contenido *application* es usado cuando el contenido del mensaje no se corresponde con ninguna de las categorías anteriores, indicado otros tipos de datos, normalmente *datos binarios no interpretados o información a ser procesada por una aplicación* (Freed y Borenstein, 1996). Entro los distintos tipos de formatos de tipo *application*, es de destacar el formato *JSON* que se ha convertido en el formato de representación más popular para intercambio de información tanto en web como para aplicaciones móviles.

2.1.3. Políticas de reemplazo

Una caché se configura con una capacidad máxima de almacenamiento. Tradicionalmente dicha capacidad se suele establecer limitando el tamaño máximo en Mega Byte (MB) que puede ocupar, aunque también existen configuraciones en las que lo que se limita es el número de objetos a almacenar con independencia del tamaño que ocupan. En cualquier caso, cuando la caché está llena, y ante la llegada de nuevos objetos que no están en ella, es necesario determinar qué objetos tienen que ser eliminados para dar cabida a nuevos. A los procesos involucrados en esta toma de decisión se les llama políticas de reemplazo.

Tabla 2.2: Factores más comunes en políticas de reemplazo

Factor	Descripción
referencia más reciente	tiempo de (desde) la última referencia al objeto
frecuencia	número de peticiones al objeto
tamaño	tamaño de los objetos Web
tiempo de búsqueda	tiempo para traer un objeto desde su origen
tiempo de modificación	tiempo de (desde) la última modificación
tiempo de expiración	tiempo a partir del cual el objeto ha caducado y puede ser reemplazado inmediatamente

Definición 3. *Se le llama política de reemplazo a los procesos involucrados en la toma de decisión de qué objetos borrar de una caché para hacer hueco a nuevos objetos.*

Las políticas de reemplazo se basan en ciertas características o factores que se emplean para decidir el objeto a borrar. En la tabla 2.2 se muestra un listado de los factores más comunes empleados en la literatura.

Una política de caché puede utilizar uno o más factores. Una vez que una política se aplica a un conjunto objetos, se obtienen una serie de propiedades y un rendimiento que depende de múltiples factores. Busari y Williamson (2001) demostraron que la función de distribución de la frecuencia de los objetos web afecta al rendimiento de una caché. Por ello, los autores concluyen que dado un conjunto de objetos, hay políticas que pueden funcionar mejor o peor en función de cómo sea la distribución de las peticiones.

Esta idea se recoge también en Wong (2006): el autor concluyó que no existe una única política que funcione mejor que las demás, y que ante un conjunto de objetos y la distribución de las peticiones, hay que seleccionar la más adecuada acorde a las características de dicho conjunto. El autor también concluyó que es mejor buscar la política de reemplazo que se adapta mejor a los datos que crear una nueva, ya que identificó hasta 50 políticas diferentes.

A continuación se listan aquellas políticas que son relevantes para los trabajos posteriores. No pretende ser una lista exhaustiva de políticas de caché. Para tal fin se puede consultar Wong (2006). Se destacan para esta tesis las siguientes:

- Least Recently Used (LRU): El factor principal que emplea es el de referencia más reciente, de tal forma que ante la necesidad de borrar de la caché, elegirá los objetos que hayan sido menos recientemente pedidos, esto es, los que hace más tiempo que no son referenciados. Esta

política es una de las más usadas en todo tipo de dominios. Asume que el tráfico presenta una localidad temporal, de tal manera que cuanto más tiempo hace que un objeto fue accedido, menos probable es que sea pedido de nuevo.

- Least Frequently Used (LFU) o Perfect LFU: Utiliza la frecuencia como el principal factor, borrando de la caché aquellos objetos que tengan la menor frecuencia de acceso, esto es, aquellos objetos que tengan menos accesos. Para ello, cada vez que llega una petición a un objeto, ésta se registra tanto si está el objeto en la caché como si no. Esta política de caché asume que la popularidad de los objetos perdura en el tiempo, y los objetos populares volverán a ser pedidos más tarde.

- LFU con envejecimiento dinámico (LFU-DA): El objetivo de este algoritmo es evitar la contaminación de la caché, esto es, que objetos antiguos con una frecuencia muy alta no puedan ser borrados. Esta variante es una mejora respecto del algoritmo anterior, ya que incorpora un factor que depende de cuándo un objeto entró en la caché. Para ello, junto a la frecuencia se introduce un factor de envejecimiento, que es un número asociado a la caché que irá cambiando con el tiempo: cuando un nuevo objeto llega a la caché ($objeto_i$), el valor que se emplea para determinar si entra o no a la caché será su frecuencia ($freq_i$) más el factor de envejecimiento de ese momento (fe), de tal forma que el factor de envejecimiento para ese objeto (fe_i) será: $fe_i = fe$. fe_i no cambiará en tanto el objeto no salga de la caché. Así el valor con el que se compara este objeto con cualquier otro será: $valor_i = freq_i + fe_i$. El factor de envejecimiento al principio vale 0, pero cuando se elimina un objeto de la caché, por ejemplo el $objeto_i$, su nuevo valor será: $fe = valor_i$. En el caso de que se eliminen varios objetos de una vez, el nuevo valor será el del menor de los objetos eliminados.

- Greedy Dual Size (GDS): Esta política usa el coste de búsqueda de un objeto así como su tamaño. Asigna un valor a todos los objetos de la caché usando los dos factores así como un factor de envejecimiento. Cuando hace falta espacio, el objeto seleccionado para ser borrado es aquel que tenga el menor valor, y su valor será el nuevo factor de envejecimiento. Es similar al anterior pero usando otros factores.

- Greedy Dual Size Frequency (GDSF): Similar al GDS, pero incluyendo la frecuencia dentro de la función a usar para asignar un valor a los objetos de la caché. Esta variante hace la política de reemplazo sensible a los cambios en la popularidad.

Una vez vistas las políticas de caché analizaremos las métricas más habituales para valorar el rendimiento de cualquier política aplicada a un conjunto de datos.

2.1.4. Métricas para medir el rendimiento de políticas de caché

Los resultados de esta tesis incluyen experimentos con cachés. Por ello, hay que analizar cómo se mide su rendimiento en la literatura. A la hora de medir el rendimiento de las políticas de caché se emplean principalmente dos métricas: *hit ratio* o tasa de acierto y el *byte hit ratio* o tasa de acierto en bytes. La primera mide el porcentaje de peticiones que son servidas desde la caché respecto del total de peticiones cacheables recibidas (fórmula 2.1). La segunda mide el porcentaje de bytes que son servidos desde la caché respecto del total de bytes que se sirven Romano y ElAarag (2011) (fórmula 2.2).

$$\frac{\sum_{i=1}^{n} h_i}{\sum_{i=1}^{n} r_i} \tag{2.1}$$

$$\frac{\sum_{i=1}^{n} b_i * h_i}{\sum_{i=1}^{n} b_i * r_i} \tag{2.2}$$

En ambas ecuaciones, h_i representa el número de aciertos (hits) para el objeto i, r_i es el número de peticiones que el objeto i ha recibido, y b_i es el tamaño en bytes del objeto i.

La elección de la métrica depende del tipo de servicio que se pretende evaluar. Romano y ElAarag (2011) notaron que ambas métricas no pueden ser optimizadas al mismo tiempo, ya que se enfocan en conceptos distintos: por un lado el número de peticiones, por el otro el volumen distribuido. Por ejemplo, optimizar la tasa de aciertos en bytes implica mantener los documentos grandes en la caché, enfoque totalmente ineficiente cuando el servicio necesita servir miles de objetos web de tamaño pequeño.

Como el foco de esta tesis está en valorar la calidad de servicio, la métrica que se va a emplear será el *hit ratio*. Estudios (Arlitt y Jin, 2000; Zhou et al., 2007) muestran que el *hit ratio* es la métrica más apropiada en contextos donde lo relevante es la calidad de servicio, y donde se pretende reducir el tiempo de respuesta de las peticiones que recibe un servidor. La misma idea

aparece en Cherkasova y Ciardo (2001), donde se demuestra que tener un alto *hit ratio* es lo más adecuado en sistemas que pretenden dar una alta calidad de servicio, ya que para grandes volúmenes de clientes permite minimizar la latencia media de las peticiones.

A continuación se presentan algunos resultados comparando distintas políticas de caché.

2.1.5. Bancos de prueba para evaluar rendimiento de políticas

Existen muchos estudios que han analizado la mejor política de caché a aplicar a muestras, y en general no existe ningún algoritmo o política que sea el mejor: dependiendo del tamaño de la caché, cómo están distribuidas las peticiones a los objetos en la muestra o el propósito de la caché, unas políticas son más ventajosas que otras y cada escenario necesita su propia solución (Wong, 2006).

En la literatura hay dos tendencias principales para analizar el rendimiento de las cachés: usar trazas reales o simular diferentes cargas y distribuciones de peticiones. Ambos enfoques son válidos aunque tienen propósitos diferentes. En el primer caso se trata de evaluar con datos reales el impacto de tomar decisiones respecto a las cachés. El segundo permite hacer elucubraciones sobre escenarios difíciles de obtener o supuestos.

Hay muchos estudios que comparan políticas de cachés, pero esta tesis no pretende ser una enumeración exhaustiva de trabajos, sino sólo aquellos que son de utilidad para los trabajos y capítulos posteriores. En concreto se va a reseñar aquellos trabajos que muestran información sobre la relación entre el espacio de trabajo (tamaño que ocupan todos los objetos en disco), el tamaño de la caché y el rendimiento de los distintas políticas de caché.

Breslau et al. (1999) analizaron la relación entre el tamaño de la caché y la política de caché más efectiva utilizando tres trazas que seguían una distribución Zipf con α entre 0,64 y 0,83 (ver sección 2.3). Su simulación consistió en lanzar las peticiones contra una caché intermedia utilizando como políticas de reemplazo LRU, Perfect-LFU y GDS. Además fueron variando el tamaño de la caché desde el 0,1 % hasta el 10 % del espacio de trabajo. El resultado de su test fue que el algoritmo GDS superó a los otros, y que el LRU presentó el menor hit ratio. En ningún caso se superó un hit ratio de 0,6.

Arlitt et al. (2000) introdujeron la política GDSF y la compararon con las políticas LFU-DA, GDS y LRU utilizando trazas reales que obtuvieron de un proxy. El tamaño de caché varió desde el 0,06 % (256MB) hasta el to 64 %

(64GB) del tamaño de trabajo. El resultado que obtuvieron fue que GDSF obtenía mejor rendimiento que las otras, que GDS era el segundo mejor, a continuación ganaba LFU-DA y finalmente el algoritmo de reemplazo LRU obtenía el menor hit ratio. El tope de hit ratio obtenido fue de 0,7, es decir, del 70 %.

Por su parte, Cherkasova y Ciardo (2001) compararon GDSF, LRU y GDS con diferentes trazas. El tamaño de caché llegó hasta el 40 % del espacio de trabajo. Los autores llegaron a la conclusión de que GDS y GDSF tienen tasas de error similares y que el LRU tiene el peor rendimiento.

Un resultado final se encuentra en Nair y Jayarekha (2010), en el cual GDSF tiene mejor rendimiento que LRU y LFU, y LRU mejora a LFU.

Estas evidencias sugieren que GDSF es la mejor política de caché, aunque en la literatura no quedan claras las características de las trazas empleadas. Además, por lo que se puede ver en las evidencias previas, es imposible obtener un alto hit rate cuando el tamaño de la caché es pequeña en relación con el espacio de trabajo.

Siendo cierto lo anterior, en una distribución de objetos con una enorme concentración de popularidad y con objetos pequeños, el papel del GDSF se vería amortiguado y potencialmente la LFU tendría un mejor comportamiento. Además de lo anterior, en las condiciones mencionadas, una caché para obtener un alto rendimiento en términos de hit ratio sólo requiere un tamaño muy pequeño, como se deriva de uno de los estudios que se han realizado en esta tesis (Zotano et al., 2015a).

2.1.6. Conclusiones

Esta sección ha servido como marco para el estudio posterior que aporta este trabajo de tesis. El comportamiento de una caché estará ligado a la función Zipf que modela las peticiones recibidas, como se verá en la sección 2.3. Entendiendo mejor cómo funcionan y se investiga en este ámbito, es posible adecuar la experimentación de este trabajo de tesis a lo que indica la literatura en el área.

El estudio de políticas de reemplazo en cachés está asentado en la literatura. Existen formas estandarizadas de medición de dichos resultados en forma de bancos de prueba y métricas como la *byte hit ratio*. El rendimiento de la caché depende también de la organización de las peticiones, por lo que el comportamiento del usuario importa. Es objetivo de las cachés el reducir el número de peticiones al sistema que genera contenidos, pero no siempre es posible. Así, la definición de cacheable limita qué contenidos pueden servirse automáticamente o hay que derivar.

2.2. Técnicas para obtención de datos

Se ha resaltado la diferencia entre objetos web y páginas web. Sin embargo, las cachés almacenan sólo objetos web. Para plantear el análisis diferenciado objetos web versus páginas web hace falta explicar qué tecnicas se recogen en la literatura para este fin. Esta sección recoge tales técnicas y son la base que permite luego calcular la presencia y forma de la función Zipf en las trazas recogidas. La sección no hace una revisión minuciosa de técnicas pero se ha tomado como referencia trabajos más extensos, como Calzarossa et al. (2016), que recoge un análisis completo de todas las técnicas para el lado del cliente y del servidor.

2.2.1. Análisis en el servidor

En la bibliografía se encuentran muchas referencias de cómo analizan los datos en el lado del servidor con diferentes fines. En esta sección se va a describir cuál es el proceso habitual en la bibliografía que se sigue para determinar la función de distribución que sigue una muestra.

Como se describe en la sección 2.3.1, hay dos técnicas para recoger los datos del servidor: el uso de proxies y utilizar los access.log del servidor. En cualquiera de los dos casos, el formato de log que suele emplearse es el formato CLF (Apache, 2004). Usando dicho formato, cada línea del log tiene el siguiente aspecto (Grace et al., 2011):

127.0.0.1 - - [10/Oct/2000:13:55:36 +0200] "GET /apache.gif HTTP/1.0"200 2326 http://apache.org "Mozilla/5.0 (Windows NT 6.1)"

Utilizando ficheros con el formato CLF es relativamente sencillo realizar análisis de muchos tipos: URLs más demandas, volumen total distribuido por un servidor, dispositivos que acceden, etc.

Muchos sistemas emplean múltiples nodos para distribuir las páginas y recursos. En estos casos es de utilidad el timestamp, ya que permiten ordenar las peticiones tal y como llegaron al servicio, con independencia de qué nodo respondió la petición. Esto es útil para hacer análisis del tráfico, así como para poder hacer simulaciones sobre el tráfico real.

Una vez que se tiene un log, el siguiente paso es el filtrado de los datos (Arlitt y Jin, 2000; Breslau et al., 1999; Nair y Jayarekha, 2010). Para ello, dependiendo del propósito del análisis, se utilizan técnicas diferentes. En el caso que ocupa esta tesis, lo que se hace es elegir sólo aquellas peticiones que son cacheables (ver sección 2.1.1).

Para determinar la función de distribución existen muchas variantes dependiendo del lenguaje, algoritmos, etc. En esta tesis se emplean dos estimadores diferentes: el estimador de Clauset (Clauset et al., 2009) que está disponible para descarga en Clauset et al. (2013), y el paquete estadístico powerlaw (Gillespie, 2014).

Ambos estimadores analizan si una muestra es una powerlaw (sección 2.3), y si es así, dan una estimación del exponente η. A partir de este valor y utilizando la ecuación 2.6 se puede determinar el valor de α, en el caso de que la distribución sea una Zipf.

Para determinar si una distribución sigue una powerlaw, los estimadores utilizan una aproximación basada en mínimos cuadráticos, calculando los posibles valores a partir de los cuales la muestra sigue una powerlaw, esto es, los posibles valores que puede tomar x_{min} (sección 2.3). Para cada uno de estos valores, se emplea el test Kolmogorov-Smirnov, el cuál seleccionará el valor que más se ajusta a una powerlaw. Para esta estimación, los algoritmos devuelven el porcentaje de aproximación y el exponente de la powerlaw.

2.2.2. Análisis en la capa del cliente

Tradicionalmente el análisis de un sitio se ha desarrollado sobre la capa de servidor, siguiendo las técnicas que han descrito en el apartado anterior. Este análisis permite obtener una perspectiva técnica de la interacción entre el usuario y el sitio web, aunque se ha empleado para extrapolar conocimiento sobre el usuario.

Sin embargo, un análisis más sofisticado puede ser llevado a cabo utilizando analítica web en la capa de cliente. Este tipo de analítica, aplicada a un sitio web, recaba información sobre las acciones del usuario cuando accede a él, cuando navega a otras páginas del sitio o cuando el usuario interacciona con sus páginas. La información obtenida es enviada a sistemas especializados en este tipo de adquisición (Google, 2016; Cloud, 2016). Entre otros datos, se manda la URL y un código de tiempo, para que a posteriori se puedan hacer análisis centrados en el contenido o en el usuario. En la bibliografía esta metodología se llama *page tagging* (Mahanti et al., 2009).

A diferencia del análisis en el servidor, que analiza todos los recursos de un sitio web, el análisis de la vista se centra sólo en las páginas web. El concepto página web engloba los objetos de negocio de un sitio. Por ejemplo, para un canal de televisión, las páginas web son noticias, vídeos, audios, fotogalerías, etc.

Existe una relación entre la vista y el servidor, ya que la petición de una página web en un navegador provoca la petición de múltiples recursos al

servidor, recursos que quedarán registrados en el access.log: se registrará la URL de la página web y las de los objetos auxiliares que estén referenciados en dicha página web.

Las métricas más básicas que se suelen obtener en cliente son las 3 siguientes: usuarios/visitantes únicos, número de visitas y páginas vistas Mahanti et al. (2009). Estas métricas están definidas como sigue (Cloud, 2016; Mahanti et al., 2009; Calzarossa et al., 2016):

Definición 4. *Una página vista es toda página web (ver definición 1) que es pedida por un usuario final. Cada vez que un usuario pide una página web un contador asociado a la URL es incrementado.*

Definición 5. *Un visitante es la persona física que entra en un sitio Web a través de un navegador.*

Definición 6. *El concepto de visitante único está relacionado con un factor temporal. Un visitante o usuario único se refiere a un visitante que visita un sitio por primera vez durante un período específico de tiempo.*

Definición 7. *Se define una visita o sesión como una serie de peticiones a recursos dentro de un marco temporal. Cuando un usuario llega a un sitio web se computa una nueva visita o sesión. Mientras esté navegando se irán sumando páginas vistas a esa sesión hasta que cierre el navegador o pasen más de 30 minutos sin ninguna actividad (no se envía ninguna página vista). En caso de haber pasado 30 minutos y enviar una nueva página vista, se computa como nueva sesión.*

El mecanismo técnico que se emplea para calcular las métricas anteriores se basa en dos identificadores que se calculan cuando un usuario accede a un sitio web: el de usuario (id_usuario en adelante) y el de sesión (id_sesion). La primera vez que un usuario accede a un sitio web, se crea una cookie (Fielding et al., 1999) conteniendo un número que será el identificar único de ese usuario. Esta cookie no caduca, así que la próxima vez que el usuario utilice el navegador, dicho valor seguirá vigente (salvo si limpia las cookies). Al tratarse de un mecanismo basado en cookies puede haber cierto sesgo ya que las cookies no se comparten entre navegadores, y al final, si un usuario usa más de un navegador, computará como más de un usuario (Calzarossa et al., 2016).

En cualquier caso, cuando un usuario accede a un sitio web, siempre se genera un id de sesión y se guarda en la cookie. Este valor tiene una duración

de 30 minutos, con lo que la cookie y por tanto el id_sesion será borrado si el usuario no hace ninguna navegación en ese tiempo.

Finalmente, y para completar el mecanismo de contabilización, cuando un usuario navega por las páginas de un portal, cada navegación implica el envío a un servicio de analítica web de la URL, el código de tiempo y las cookies, y por tanto el id_usuario y el id_sesion. El servicio de analítica recibe las peticiones y almacena la información para realizar informes o para auditar audiencias.

Respecto de los servicios de analíticas web, hay muchos servicios disponibles, algunos gratuitos y otros de pago. Los sitios web suelen utilizar más de uno para contrastar la información que se genera. En el mercado español, la práctica habitual es la de utilizar Comscore, Adobe Omniture y/o Google Analytics. A la hora de auditar y validar los datos que recogen estos servicios, en España hay una empresa a la que se le ha encomendado dicha tarea, OJD (2016). Esto garantiza la calidad de los datos recogidos en los sistemas de analítica de cliente.

2.2.3. Conclusiones sobre técnicas de obtención de datos

Como en otras áreas, un sistema (servicio web) puede contemplarse desde ópticas distintas. En el caso de la web, un servicio se puede analizar desde la perspectiva del cliente o desde la del servidor. El hecho de elegir una perspectiva u otra nos puede llevar a diferentes resultados. Cada una de las aproximaciones tiene ventajas y desventajas. Mahanti et al. (2009) analiza ambos enfoques encontrando que las principales diferencias son:

- La analítica del lado de cliente no permite medir el volumen de tráfico, ya que se enfoca en elementos de negocio y en qué consumen los usuarios. El lado de servidor, al enfocarse en la parte más técnica, sí que lo permite.

- La analítica del lado de cliente permite hacer un seguimiento del usuario gracias al uso de cookies, mientras que en el lado de servidor dicho seguimiento se hace con la Internet Protocol (IP) del usuario. El uso de de la dirección IP tiene el hándicap de que los proxies pueden hacer que muchos usuarios la compartan y haya un sesgo importante. Por su lado, las cookies no siguen al usuario a todos los dispositivos y navegadores desde los que un usuario pueda acceder, con lo que tampoco está libre de sesgo.

- El análisis de servidor es afectado por las cachés y no permite distinguir a humanos de robots.
- Finalmente, sólo con analíticas de cliente se puede obtener información tales como el sistema operativo del usuario, capacidades de los navegadores, etc.

2.3. Funciones Zipf y modelado de tráfico web

De acuerdo a Mahanti et al. (2013) una *power-law* (función exponencial) es una función que tiene la siguiente forma:

$$f(x) = kx^{-\eta} \tag{2.3}$$

donde k y η son constantes positivas y donde η es el exponente de la función exponencial. Cuando se aplica logaritmo en ambos lados se obtiene:

$$log(f(x)) = -\eta log(x) + log(k) \tag{2.4}$$

La fórmula 2.4 corresponde a la función que describe una línea recta con pendiente $-\eta$ (ver figura 2.1).

Un conjunto de datos se dice que sigue una función de distribución power-law cuando la distribución es cercana a una power-law para valores altos (Mahanti et al., 2013; Mitzenmacher, 2004): $f(x) \sim kx^{-\eta}$. Para ser una power-law se asume que no tiene que existir una correspondencia exacta en los valores bajos, pero sí a partir de cierto valor. El símbolo $\sim$ cobra especial importancia ya que implica un comportamiento asintótico: el conjunto de datos seguirá la distribución exponencial a partir de un cierto valor x. Al conjunto de datos a partir x se le denomina como la cola de la distribución (Mahanti et al., 2013).

Una Zipf (1949), por su parte, es una distribución power-law que fue usada por primera vez en linguística para modelar las ocurrencias de las palabras en ingles. Formalmente, una *Zipf es una distribución discreta definida en el dominio del ranking versus frecuencia, de tal manera que cuando se ordenan los items de forma descendente por ranking de popularidad, entonces la frecuencia de dicho item es inversamente proporcional al ranking del item* (Mahanti et al., 2013). Según dicha definición, la fórmula de una Zipf se puede escribir así:

$$f(r) \sim kr^{-\alpha} \tag{2.5}$$

donde k y α constantes, r es el ranking y el valor de α es cercano a 1. Dos definiciones se emplean habitualmente en la literatura en función del valor de α:

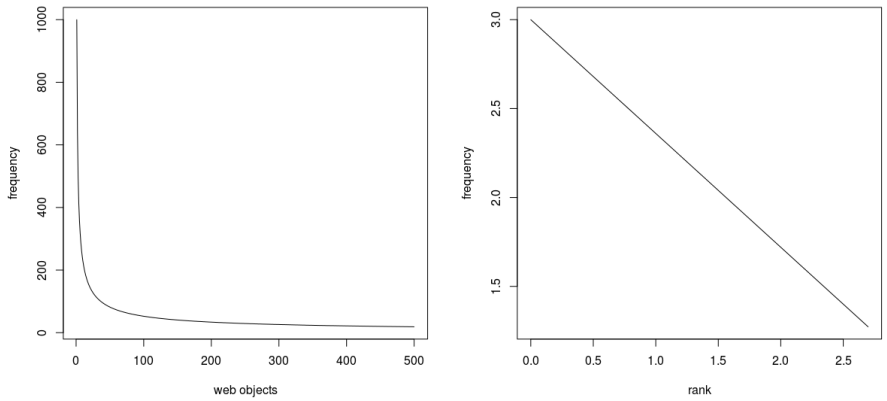


Figura 2.1: Función power-law (izquierda) y su transformación tras aplicar una escala logarítmica (derecha)

Definición 8. *Una distribución se dice que es una Zipf estricta si el valor de α es casi igual a 1.*

Definición 9. *Una distribución es de tipo Zipf cuando α es distinto de uno.*

Al ser una Zipf una power-law, la fisonomía de línea recta aparece cuando se dibuja la distribución al aplicar logaritmo en los dos ejes. No obstante, en conjuntos de datos reales suele aparecer una forma degenerada donde sólo una porción de los datos siguen la distribución Zipf.

En la figura 2.1 se puede ver un ejemplo donde la función power-law está en la izquierda y la transformación tras aplicar logaritmo aparece en la derecha. Como se puede apreciar, al aplicar los logaritmos se muestra una línea.

En Mahanti et al. (2013) se muestra cómo se relaciona el exponente de una power-law (ecuación 2.3), η, y el exponente de la función Zipf correspondiente α (ecuación 2.5):

$$\eta - 1 = \frac{1}{\alpha} \tag{2.6}$$

La equivalencia mostrada en la fórmula 2.6 es útil para determinar el valor de α en un conjunto de datos, ya que es sencillo calcular el valor de η, y a partir de él se calcula α.

En el dominio de la Web, una Zipf se interpreta como la distribución de las peticiones de los usuarios que llegan al servidor. Como las peticiones se distribuyen sobre un número finito de objetos, y como las peticiones de los clientes se pueden considerar como eventos independientes, la popularidad de

un objeto web se caracteriza por su probabilidad. De esta forma, la función de distribución de la popularidad se define en Shi et al. (2006) como una redefinición de una Zipf, a través de la fórmula $P_i = \frac{C}{i^\alpha}$, donde C es una constante, i el i_{esimo} objeto más popular y α el parámetro de la Zipf.

El valor del exponente α es un indicador del grado de homogeneidad de la muestra. Así, cuanto mayor sea el valor, menor será la homogeneidad y menos objetos serán necesarios para llegar a un cierto porcentaje de las peticiones. Al contrario, cuanto menor sea el valor de α, mayor será la homogeneidad y mas objetos serán necesarios para alcanzar dicho porcentaje de peticiones. Dada una muestra cuya frecuencia de aparición se puede describir con una Zipf, se dirá que cuanto mayor sea α, menor será la homogeneidad de la muestra y menos objetos serán los que aparezcan con mayor frecuencia. Cuanto menor sea α, mayor será la homogeneidad y mayor el número de objetos que aparecen con mayor frecuencia. En el ámbito de la web, esto se puede reformular como la propiedad 1.

Propiedad 1. *El valor del exponente α es un indicador del grado de homogeneidad de la muestra que sigue una función Zipf. Así, cuanto mayor sea el valor, menor será la homogeneidad y menos objetos serán necesarios para llegar a un cierto porcentaje de las peticiones. Al contrario, cuanto menor sea el valor de α, mayor será la homogeneidad y mas objetos serán necesarios para alcanzar dicho porcentaje de peticiones.*

Aunque las power-laws son definidas en el domino de los números reales, en los sitios webs las peticiones se contabilizan como números naturales. Por ello la línea recta de la Zipf es sólo una aproximación donde los valores en los primeros rankings (los más populares) y en los últimos (los menos populares) pueden aparecer fuera de la línea cuando ésta se dibuja.

Finalmente, los algoritmos que se emplean para determinar si una distribución es una Zipf (Clauset et al., 2009) también suelen calcular un valor x_{min}, que es el ranking a partir del cuál un conjunto de datos comienza a comportarse como una Zipf. Este valor ayuda a hacer estimaciones y a descartar elementos que tienen un comportamiento fuera del modelado (Clauset et al., 2009, 2013).

Definición 10. *Denotamos por x_{min} el ranking a partir del cuál un conjunto de datos comienza a comportarse como una Zipf.*

En la siguiente subsección se discutirán las consecuencias y beneficios para un webmaster de que las peticiones a un servicio sigan una distribución Zipf.

2.3.1. Valores de Zipf y técnicas empleadas en la literatura sobre modelado de tráfico web

La existencia de distribuciones que siguen una Zipf esta bien soportada por resultados empíricos. Para conseguir estos resultados es necesaria la adquisición de conjuntos de datos en condiciones reales. Para tal fin se suelen emplear dos técnicas: obtener y analizar los *access.log* (fichero que almacena todas las peticiones que llegan a un servicio Web) de un sitio web o bien utilizar un elemento intermedio (proxy) dentro de una red que registre todas las peticiones entre clientes-servidores durante un período de tiempo. En la literatura se describe que el uso de una u otra técnica puede provocar distintos resultados para una misma muestra (Breslau et al., 1999; Almeida et al., 1996).

Tras la adquisición de los datos, esto es, de las peticiones que se han realizado por parte de los usuarios, estos son analizados para poder estimar la presencia de una Zipf, y en su caso cuál es el valor de α, el número de objetos únicos que presenta la muestra y el tamaño de la muestra.

En lo que sigue se puede ver que la mayoría de las trazas que se han analizado provienen de proxies que recuperan las peticiones en centros controlados o que mezclan peticiones de múltiples sitios. Cuando se pretende analizar un website, el hecho de recolectar los datos utilizando proxies puede llevar a inexactitudes ya que no todas las peticiones son recogidas. En ese caso es mejor el uso de los logs del sitio. De todas formas, aunque puedan darse algunos sesgos, se pueden inferir algunas propiedades a partir de logs incompletos, incluso una aproximación de α, aunque inexacta.

El primer resultado relevante que aparece en la literatura fue publicado por Glassman (1994). En este estudio se utilizó un proxy para obtener todas las peticiones externas en las oficinas de Digital Equipment Corporation en Palo Alto, California. Para este estudio se recogió una muestra de 100.000 peticiones que fueran capturadas, y tras su análisis, el autor concluyó que las trazas seguían una distribución Zipf con un valor de α igual a uno. Un año después, un resultado similar fue encontrado por Cunha et al. (1995) utilizando la misma técnica en la Universidad de Boston. Concretamente el valor de α fue igual a 0,986. Como se puede ver, sendas Zipf estrictas fueron el resultado de ambos experimentos.

Almeida et al. (1996) analizaron los *access.log* del servidor Web de la Universidad de Boston, encontrando que la distribución de peticiones sobre los objetos web de dicho servidor seguían una distribución Zipf con un valor de α igual a 0,85. En este caso el número de peticiones fue de 80.518 sobre un total de 4.471 objetos. En este artículo los autores también concluyeron que

la distribución de accesos utilizando la información del website y utilizando la técnica del proxy da resultados diferentes. Esto es debido a que cuando se emplea un proxy, sólo una porción de las peticiones que llegan al servidor son capturadas, y de ahí que se de un valor distinto.

Más tarde, Nishikawa et al. (1998) de la empresa Hitachi analizaron el proxy de una instalación que incluía 2.000.000 de peticiones a más de un sitio Web. Los autores obtuvieron una distribución tipo Zipf con un valor de α de tan solo 0,75.

Breslau et al. (1999) analizaron las trazas recogidas por proxies en seis instituciones académicas. Tras este estudio concluyeron que una distribución tipo Zipf aparecía en todas ellas, y el valor de α encontrado estuvo acotado entre 0,64 y 0,83. El número de peticiones recogidas varió entre 1.776.409 y 4.815.551, y el lapso de tiempo de recogida fue de entre 1 día hasta 3 meses.

Por su parte, Roadknight et al. (2000) analizaron los logs de 5 servidores de caché y concluyeron que cuanto mayor es el volumen de peticiones que se registran, mayor es la estabilidad del valor de α. Los autores acotaron su valor entre 0,5 y 0,9. También encontraron que para la determinación de una valor preciso de α se requiere de largos volúmenes de datos, es decir, de un volumen alto de peticiones. Además concluyeron que las deducciones hechas utilizando volúmenes pequeños de peticiones lleva a errores de cálculo. Sugieren que en cualquier caso se utilicen conjuntos grandes de datos. Finalmente, también dedujeron que el valor de α depende de la posición en la que se encuentre la caché dentro de la topología de la red.

Arlitt y Jin (2000) analizaron el sitio del *Campeonato del Mundo de la FIFA 1998*, el cual recibió en menos de un mes un total de 1.352.804.107 peticiones desde cliente, obteniendo que la distribución de los objetos web mostraban 3 regiones diferenciadas:

1. Los primeros 100 objetos recibieron el 61 % del volumen total de peticiones, mostrando un valor de α igual a 0,25.

2. La segunda sección (20.000 objetos) mostró un valor de α igual a 1,92.

3. El resto mostró un valor de α de 14,7.

Cuando restringieron el análisis a sólo ficheros HTML obtuvieron una distribución Zipf con un valor de α igual a 1,16.

Mahanti et al. (2000) analizaron la carga completa de 3 proxies concluyendo que presentaban un valor de α entre 0,74 y 0,84. El volumen de peticiones utilizado en este experimento fue entre 7.310.038 a 24.560.611, con un promedio diario de peticiones entre las 162.445 y las 818.687.

El contenido multimedia también ha sido estudiado y empleado por diversos autores con el objetivo de determinar si las distribuciones de este tipo de contenido siguen o no una Zipf.

Ese es el caso de Chesire et al. (2001) quienes en 2001 recogieron y analizaron el volumen de peticiones que durante una semana generó la actividad de los clientes de la Universidad de Washington. Para ello, utilizaron un proxy, con el cual obtuvieron información de 4.786 sesiones que pidieron 23.738 objetos multimedia de entre 866 servidores de Internet. Encontraron que la distribución de la popularidad tenía una forma de Zipf con un valor de α igual a 0,47. Además concluyeron que el 78 % de los objetos sólo se accedieron una vez (en la terminología one timers), que sólo el 1 % fueron accedidos por 10 o más sesiones y que únicamente 12 objetos fueron pedidos por más de 100 sesiones.

Challenger et al. (2004) analizaron los logs del site de los Juegos Olímpicos de Nagano 2004, utilizando 200.000 peticiones al servidor y encontrando que la distribución obedecía a una Zipf estricta.

En 2006 se publicó un artículo de Krashakov et al. (2006) que recogía un resumen de las trazas aparecidas hasta ese momento. Además, los autores incluyeron sus resultados tras analizar un conjunto de logs utilizando proxies. El análisis consistió en determinar si seguían una Zipf, y si una variación de la Zipf era mejor para describir el comportamiento. La fórmula que introdujeron es: $fr = \frac{b}{(c+r)^{\alpha}}$. Este variación consistió en sumar una constante c al ranking en la fórmula general de la Zipf. Los autores concluyeron que la introducción de esta constante hacía α más estable a lo largo del tiempo, obteniendo un valor de α igual a 1,02 $\pm$ 0,05 para sus trazas.

Gonzalez-Canete et al. (2007) hicieron un estudio en el que descomponen una Zipf según el tipo de contenido (ver sección 2.1.2) y analizan si cada una de las distribuciones por tipo de contenido también sigue una Zipf. Para ello los autores analizaron las trazas obtenidas en un proxy en el Research Triangle Park (North Caroline, USA) con las peticiones realizadas durante cuatro días, desde el 7 al 11 de Junio de 2004. Los autores encontraron que la distribución de peticiones de esos cuatro días sigue una Zipf con un valor de α igual a 0,64.

Los autores, a continuación, dividieron el log por tipo de contenido, encontrando que las cinco nuevas distribuciones también siguen una Zipf. En concreto, los autores encontraron que la distribución de las peticiones a objetos de tipo *application* mostraba un valor de α igual a 0,78, las de tipo *audio* igual a 0,4, las de tipo *image* igual a 0,46, las de tipo *text* igual a 0,54 y finalmente las de tipo *video* mostraban un valor de 0,95.

Finalmente los autores obtuvieron algunas estadísticas sobre la densidad

de los tipos de contenidos en el log, encontrando que los objetos de tipo image eran los más demandados, suponiendo hasta el 75 % de todas las peticiones, a continuación los de tipo text alcanzaron el 15 %, y los otros tipos restantes supone tan sólo el 10 % del total de peticiones.

Servicios Web altamente demandados tales como *Facebook*, *Youtube* o *Wikipedia* también han sido objeto de análisis. Gill et al. (2007) hicieron un análisis del consumo que se hizo del servicio de *Youtube* en la red del campus de la Universidad de Calgary durante un período de tres meses. Después de analizar 24.668.616 peticiones, los autores encontraron que las peticiones de vídeo bajo demanda seguían un distribución tipo Zipf con un valor de α igual a 0,56.

Urdaneta et al. (2009) publicaron un artículo que analizaba los logs de la *Wikipedia*. Tras procesar 25,6 miles de millones de peticiones de lectura, encontraron que sólo las 20.000 páginas más demandas seguían una distribución Zipf. El resto de páginas tenían unas frecuencias menores de las que se esperaría si la distribución completa sigue una Zipf. Al analizar las peticiones de escritura encontraron que en este caso sí que se daba una Zipf con un valor de α igual a 0,64.

Finalmente, Huang et al. (2013) estudiaron el sistema de múltiples niveles de caché de *Facebook*, y encontraron que las peticiones seguían una Zipf tras analizar 77 millones de peticiones a 1,3 millones de fotos. Los autores indagaron sobre cómo variaba α en las diferentes capas de caché dentro de la arquitectura Web: se emplea una caché en la capa de cliente, además de varias capas dentro de la arquitectura del servidor. Tras el análisis, se encontró que en todas las capas la distribución de las frecuencias sigue una Zipf, incluyendo la capa de cliente. Además notaron que α se va reduciendo a medida que las capas están más lejos del cliente, de tal manera que cuanto más cerca al cliente, mayor es el valor de α.

Este resultado es coherence con el efecto *Trickle Down* que fue descrito por Doyle et al. (2002). Este efecto se da en sistemas con múltiples capas, donde cada capa implementa una caché. Consiste en un aplanamiento de la distribución de probabilidad de una muestra a medida que se va profundizando en las capas, de tal manera que el sesgo que se ve en una capa disminuye en la siguiente, y así sucesivamente.

En la tabla 2.3 se ha hecho un resumen de las diferentes evidencias encontradas sobre la existencia de Zipf en servicios Web.

Tabla 2.3: Resumen de evidencias de la existencia de Zipf

Autores	α	Año	Trazas	Técnica
Glassman et al.	1	1994	1	Proxy
Cunha et al.	0,986	1995	1	Proxy
Almedia et al.	0,85	1996	1	Servidor Web
Nishikava et al.	0,75	1998	1	Proxy
Breslau et al.	0,64 - 0,83	1999	6	Proxy
Roadknight et al.	0,5 - 0,9	1999	5	Proxy
Artlitt and Jin	1,16	2000	1	Servidor Web
Mahanti et al.	0,74 - 0,84	2000	3	Proxy
Chesire et al.	0,47	2001	1	Proxy
Challenger et al .	1	2004	1	Servidor Web
Krashakov et al.	$1{,}02 \pm 0{,}05$	2006	5	Proxy
Gill et al.	0,56	2007	1	Proxy
Urdaneta et al.	0,64	2009	1	Servidor Web
Huang et al.	1 in client layer	2013	1	Servidor Web

2.3.2. Implicaciones de la existencia de una Zipf en el diseño de una estrategia de caché

En la subsección anterior se describió qué es una Zipf. En esta se va a analizar qué implica que los objetos web de un servidor sigan una distribución Zipf. Todas las consecuencias se van a centrar en el dominio de la Web.

Cuando un conjunto de datos sigue una Zipf la popularidad se convierte en un factor a tener en cuenta. Si el conjunto de datos pertenece al dominio de la Web y al lado de servidor, la frecuencia se convierte en un factor clave en las políticas de cache que se tengan que utilizar para optimizar el hit ratio. Además de lo anterior, también es una consecuencia que el hit ratio máximo en caso de usar una cache se puede predecir. Esto sólo es posible si se sigue una Zipf. Como consecuencia de la propiedad 1, el valor identificado de α influye en la cantidad de objetos que hay que recordar. Por ello, las consecuencias más relevantes en el dominio web son la importancia de la frecuencia y la capacidad de predecir el hit ratio teórico máximo de una muestra.

Como precedente, las consecuencias de que las peticiones de los usuarios sigan una distribución Zipf han sido analizadas en la literatura. En Breslau et al. (1999) se hace un estudio de 6 trazas obtenidas utilizando un proxy. En el artículo se estudia además la relación entre el hit ratio y el tamaño de la cache para esas distribuciones. Las conclusiones principales de dicho estudio

son:

- Para una cache de tamaño infinito, el hit ratio, como función del número de objetos web y de peticiones de usuarios, crece de forma logarítmica. Esto significa que el hecho de que haya más peticiones no implica necesariamente que se incremente de forma lineal el hit ratio sino de forma logarítmica.
- El valor del hit ratio en función del tamaño de caché, presenta una función logarítmica. Es decir, que el hit ratio es proporcional al logaritmo del tamaño de caché.
- En porcentaje, entre el 25 y el 40 % de los objetos Web reciben el 70 % de las peticiones.
- Al respecto de las políticas de cachés en servidor, las pruebas realizadas por los autores muestran que el algoritmo Perfect-LFU funciona mejor que el LFU, GD o el In-Cache-LFU, cuando se compara según la métrica del byte hit-ratio. Respecto el hit-ratio, el Greedy Dual (GD) es el que funciona mejor para cachés pequeñas mientras que el Perfect-LFU es mejor para cachés grandes.

Por su parte, Shi et al. (2006) concluyeron a partir de sus experimentos que en el caso de una distribución Zipf y dado un tamaño de caché, la política de caché LFU es la que da mejores resultados ya que prima la frecuencia de los objetos frente a otros criterios, al igual que hace la Zipf.

Además del resultado anterior, los autores introdujeron el concepto de *estabilidad*: es un método matemático para determinar el número de peticiones necesarias para obtener un valor estable del exponente α. Ya que una Zipf es una función asintótica, α puede variar dependiendo el número de peticiones analizadas. La fórmula introducida por los autores permite determinar el número mínimo de peticiones a analizar para conseguir un resultado preciso.

Respecto de la capacidad de predicción, esto es, la capacidad de determinar el hit ratio esperado dado un tamaño de caché, los mismos autores (Shi et al., 2005) publicaron la siguiente fórmula que depende de α, del número de objetos únicos y del tamaño de la caché que se va a emplear con una distribución que sigue una Zipf:

$$k = N * h^{\frac{1}{1-\alpha}} \quad si \quad \alpha \neq 1 \qquad y \qquad k \approx e^{h*lnN} \quad si \quad \alpha = 1 \tag{2.7}$$

N es el número de objetos distintos, h es el hit ratio, $\alpha \leq 1$ y k es el tamaño de la caché. Como se puede deducir de la fórmula, no se contempla el hecho

de que $\alpha > 1$ ya que no es un resultado consistente en la literatura. De cualquier forma, desde un punto de vista industrial, es relevante el resultado puesto que permite pronosticar el efecto de incrementar el tamaño de una caché.

Nair y Jayarekha (2010) analizaron el uso de cachés en servidores multimedia, concluyendo que para este tipo de objetos también se da una distribución basada en Zipf. Además concluyeron que la política de caché LFU basada en la frecuencia es la que mejor resultados ofrece en términos de hit ratio.

Una consideración final respecto del cacheo web se puede encontrar en Mahanti et al. (2013): en al artículo se afirma que el contar con una distribución de popularidad no uniforme para los objetos web de un sitio, así como el hecho de que esta distribución siga una Zipf, permite a los diseñadores de soluciones estimar el tamaño de la caché para obtener el nivel de hit ratio deseado.

Como conclusión, las consecuencias principales de ser Zipf, en el lado de servidor, son la relevancia de la frecuencia y la capacidad de predicción. En la siguiente subsección se muestran qué evidencias existían de la existencia de Zipfs en la bibliografía. En este listado no aparecen los resultados que hemos conseguido por nuestra parte.

2.3.3. Conclusiones sobre trabajos previos en Zipf

Las principales conclusiones que se pueden extraer tras analizar las evidencias sobre Zipf en la literatura son:

1. Es cierto que existen evidencias de trazas o logs que no siguen una Zipf, como (Almeida et al., 1998), pero estas excepciones suelen tener un volumen bajo de peticiones. Si atendemos a las conclusiones de Shi et al. (2005), hace falta una masa crítica para que se manifieste el comportamiento Zipf, por lo que en logs pequeños se suelen encontrar este tipo de excepciones. De hecho, la distribución Zipf aparece en casi todos los estudios que se han hecho. La implicación de este hecho, es que la popularidad de los objetos web sigue un patrón común, una Zipf, con independencia del método utilizado para obtener los datos, incluso cuando estos provienen de múltiples dominios mezclados. Es por ello por lo que la popularidad o la frecuencia de acceso tiene que ser considerado como factor relevante en el diseño de políticas de caché.

2. El valor de α obtenido empíricamente a partir de las trazas está acotado entre 0,6 y 1 en la mayoría de los trabajos (Breslau et al., 1999; Shi et

al., 2005; Nair y Jayarekha, 2010). Existen algunas excepciones tales como la distribución de los ficheros HTML en las trazas obtenidos de la *FIFA* comentada anteriormente.

3. El valor de α para un sitio Web puede diferir dependiendo de la técnica empleada para obtener los datos: el valor obtenido utilizando los logs de un sitio Web es diferente del que se obtiene utilizando un proxy debido a que el conjunto de peticiones varía (basta con que haya un usuario que acceda al sitio sin pasar por el proxy). De hecho, si nos fijamos en la tabla 2.3 se puede observar que α es mayor cuando se analizan los logs de los servidores que cuando se emplean proxies.

4. En cuanto al número de peticiones, cuanto mayor sea, más preciso será el valor calculado de α, pero si el número de peticiones considerado es demasiado alto, puede pasar que la cola de la distribución tenga unos valores menores de los que predice la Zipf.

5. El valor de α depende de la posición en la que se ponga el proxy o la caché dentro de la red, o de la capa que se esté analizando dentro de la arquitectura Web (Roadknight et al., 2000; Huang et al., 2013).

6. Finalmente no todas las peticiones son analizadas para determinar si una distribución sigue o no una Zipf. Cuando se analiza un log, sólo las peticiones HTTP con un código de respuesta 20X o 40X (Breslau et al., 1999; Roadknight et al., 2000; Nair y Jayarekha, 2010) son de interés, ya que la inclusión de los código de respuesta HTTP 3XX podrían suponer una doble contabilización de los mismos objetos Web. Hay que tener en cuenta que un código 3XX es un redirect y que un cliente, tras recibir dicha respuesta realizar la llamada al recurso apuntado (Fielding et al., 1999).

2.4. Conclusiones generales

El análisis de cómo los usuarios emplean un sistema es vital para poder modelar su comportamiento y así mejorar los sistemas que proporcionan dichos servicios. En el caso de la web, el análisis de los usuarios, esto es, qué recursos piden y con qué frecuencia, permite extrapolar el comportamiento que se va a detectar en el servidor, y por tanto permite optimizar y mejorar los servicios.

En la web el comportamiento de los usuarios se suele modelar utilizando como variables las páginas web y los usuarios, de tal manera que se analiza

qué páginas son accedidas por cada usuario. Así se puede hacer inferencias en cliente, utilizando las técnicas de *page tagging*, o bien capturando las peticiones que desde dichas páginas se generan a recursos web, técnica utilizada para el servidor. Este modelado tiene también un valor adicional: permite determinar si el uso del servicio ha cambiado y por tanto determinar mecanismos reactivos ante dichos cambios. Un ejemplo muy extendido de lo anterior es el análisis de los usuarios en un sitio web, y el impacto de su comportamiento en mecanismos de escalado como son las cachés.

En líneas generales, y como se ha podido ver en la sección 2.3, hay un consenso en la bibliografía sobre la existencia de Zipf en el dominio de la web. El hecho de ser una Zipf tiene dos grandes ventajas: por un lado la predictibilidad y por el otro la relevancia de un factor, la frecuencia, que será crucial a la hora de mejorar la calidad del servicio. Sin embargo, las evidencias son antiguas, y debería revisarse cómo de homogénea son las distribuciones en los servicios actuales. De hecho, en la bibliografía no se contempla que α puede ser mayor que uno, ni se ha estudiado qué implicaciones tendría si esto ocurriese. Tras analizar las evidencias hay algunas inexactitudes, dudas o elementos que requieren una revisión. En concreto se ha detectado:

1. La mayoría de las trazas provienen de instituciones académicas, por lo que consideramos que es necesario una mayor variedad de trazas que provengan de distintos sectores, incluyendo sites para la venta de objetos, compañías de distribución multimedia, etc.

2. En la literatura se ha hecho un uso intensivo de proxies debido a las restricciones en el acceso a los logs de sitios Web. Pero el uso de proxies puede llevar a inexactitudes o errores (Almeida et al., 1996). Por ello podría ser relevante un estudio más intensivo en el cuál se empleen sobre todo logs de sitios Web.

3. El volumen de peticiones en la inmensa mayoría de estudios no es equivalente al volumen que se da en la actualidad. Hay que tener en cuenta que el número de usuarios conectados a Internet ha crecido exponencialmente, y por ello sería necesario validar que los servicios Web siguen presentando distribuciones tipo Zipf, y si el valor de α sigue estando por debajo de 1.

4. En ningún caso se ha estudiado ni hecho mención a la tecnología usada para el desarrollo de los sitios Web, y quizá hay correlaciones no estudiadas entre la tecnología y la distribución de las peticiones.

Como cada una de las perspectivas da una visión diferente, siendo interesante ver si se pueden correlacionar comportamientos en la capa de la vista,

con sus consecuencias en la capa de servidor, algo que la literatura no ha considerado todavía.

En esta tesis se abordan varias de las cuestiones que han ido surgiendo en este capítulo. La primera de ellas, que será tratada en el siguiente capítulo, es si los sitios web actuales siguen distribuciones de frecuencia para sus objetos web con una forma de Zipf .

Capítulo 3

Evidencias de Zipf en servidor en trazas actuales

En el capítulo anterior se han mostrado evidencias sobre la existencia de distribuciones de frecuencia que siguen una Zipf en el domino de la Web. Como se ha comentado previamente, una Zipf está caracterizada por el exponente de la función exponencial, α (ver sección 2.3). En la bibliografía hay un consenso de que las Zipf es la distribución habitual en los sitios web, y el valor de α, calculado a partir de evidencias empíricas, está entre 0,6 y 1.

En este capítulo, siguiendo con lo que se comentó en la sección 2.4, se va a validar si la distribución Zipf está presente en la actualidad en el dominio de la web, y si es el caso, si el valor α sigue estando acotado a la Zipf estricta. Además se va a evaluar si existe alguna correlación entre el valor de α con la tecnología del sitio web o con el sector al que está dirigido el sitio web.

Los datos y resultados de este capítulo han sido publicados en Zotano et al. (2015b) (ver apéndice A).

3.1. Validación Zipf en trazas reales

Para validar la existencia o no de Zipf en trazas actuales, se han recogido los access.log de 16 sitios web. Estos logs contienen la información descrita en 2.2.1, y por tanto la IP del cliente, el método usando en la petición HTTP, la URL y el timestamp. Por temas legales la información de las IP se eliminan de los logs, pero de acuerdo a la bibliografía (Breslau et al., 1999; Roadknight et al., 2000; Nair y Jayarekha, 2010), no se va a producir ningún sesgo al no ser una información relevante. Los logs serán procesados cómo se describió en la sección 2.2.1.

Obtener una amplia variedad de logs es una tarea compleja por motivos

Tabla 3.1: Logs analizados

Sector	Sitio	Período	Peticiones	Technología
Tele y Radio	rtve.es	1 día	130 millones	PHP, J2EE
Noticias	eldiario.es	1 día	8.961.959	HTML
B2B y B2C	bodaclick.com	1 día	13.987.028	PHP
B2B deportes	ebuops.com	1 mes	330.043	Struts2
Universidad	ucm.es	1 mes	8.379.230	PHP
Universidad	fdi.ucm.es	3 meses	3.080.476	ASP.Net
Investigación	grasia.fdi.ucm.es	3 meses	74.023	Drupal
Investigación	npcassoc.org	3 meses	328.511	Wordpress
Juego	otogami.com	3 meses	2.547.172	J2EE
Ayuntamiento	pedro-munoz.es	4 meses	4.302.641	Drupal
Ayuntamiento	pdmmadridejos.es	1 mes	221.062	Drupal
Ayuntamiento	cobisa.es	2 meses	42.753	Drupal
Polideportivo	deportesmota.es	5 meses	384.416	PHP
Educación	institut-francais-israel.org	13 días	168.753	Wordpress
Herramienta	jira.rtve.int	20 días	1.246.808	Tomcat
Herramienta	wiki.rtve.int	20 días	3.469.508	Tomcat

comerciales y de privacidad. Las empresas son muy reticentes a facilitar la información de cómo funcionan sus sistemas. Para esta tesis hemos contactado con muchos webmasters y hemos conseguido obtener los logs de 16 servicios web. En la tabla 3.1 aparece la lista de dominios de los cuales hemos recogido los logs, todos ellos recopilados en el año 2014. Como se puede observar en la tabla, para cada log se indica el dominio, el número de peticiones que incluye, el período de adquisición de peticiones, el sector al que pertenece la actividad del servicio y finalmente la tecnología que emplea el servidor.

En la tabla 3.1 se puede observar que el volumen de trazas que se ha recogido supera a la de casi todos los trabajos anteriores. Hay que prestar especial atención a rtve.es, la web del la televisión pública española. Este servicio recibe muchas peticiones, de hecho más de 100 millones de ellas al día, distribuyendo una gran variedad de tipos de contenidos. Los contenidos principales que distribuye son noticias y contenido multimedia. Estos contenidos se sirven a través de portales, aplicaciones móviles, radio en directo, canales de televisión en directo, etc. Además de rtve.es, el sector de noticias y multimedia está representado por eldiario.es, otro portal informativo, útil para poder hacer comparaciones con rtve.es ya que son del mismo sector.

Los demás servicios pertenecen a otros sectores: bodaclick.com y ucm.es son de interés por el gran volumen de peticiones que reciben; jira.rtve.int destaca por ser un servicio basado en login de usuario empleado como herramienta de gestión de tareas en RTVE y wiki.rtve.int, sistemas basado en login utilizado para compartir documentos. Finalmente, el resto no recibe tanto volumen de peticiones, pero son de utilidad para enriquecer el análisis.

Tras aplicar las técnicas descritas en la sección 2.2.1, es decir, tras filtrar los logs y contabilizar para cada url su frecuencia, se genera un fichero con la distribución de frecuencia por objeto web. A continuación ordenamos los objetos descendentemente por frecuencia, aplicamos logaritmos y pintamos la distribución. Actuando así se obtienen las imágenes que compone la figura 3.1. En ella aparecen doce imágenes, cada una de ellas es la imagen gráfica de un sitio web. En el eje x aparece el logaritmo del ranking, mientras que en el eje y tenemos el logaritmo de las frecuencias. Los puntos representan los valores y por debajo se ve una línea que tiende a seguir a los puntos. Esta es la línea teórica que seguiría una Zipf. Tal y como se comentó en la sección 2.3, se puede ver que la línea recta aparece durante la mayoría de la traza, y que no se ajusta ni al principio ni al final. Esto es coherente con la bibliografía (Mahanti et al., 2009).

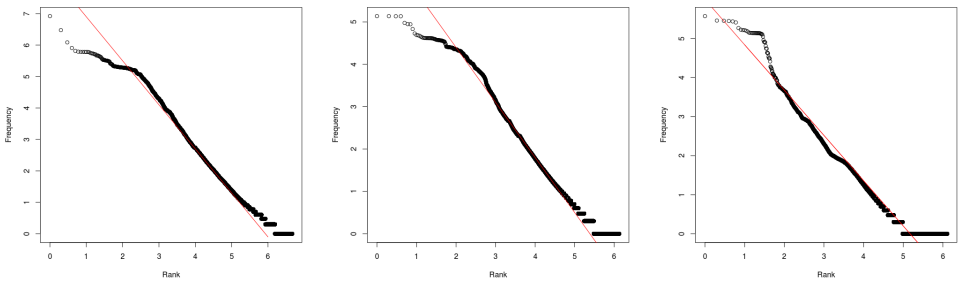
Para determinar que las distribuciones son realmente Zipfs es necesario analizar las trazas con métodos matemáticos. Por ello se emplean los estimadores para determinar si siguen una powerlaw. Los estimadores (ver sección 2.2.1) permiten calcular el valor de η y de x_{min}. Utilizando la fórmula 2.6 se calcula el valor de α para la muestra.

Siguiendo el proceso descrito, en la tabla 3.2 aparece el valor de α y x_{min} para cada uno de los 16 logs.

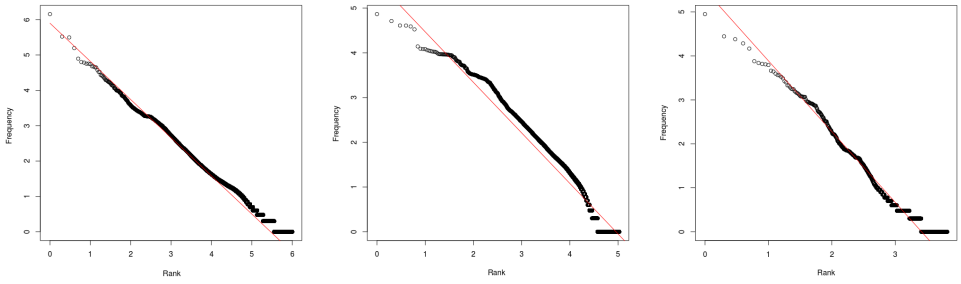
3.2. Análisis y conclusiones sobre los resultados

Una vez presentados los datos a analizar y los resultados del análisis, en esta subsección se van a discutir e interpretar dichos resultados.

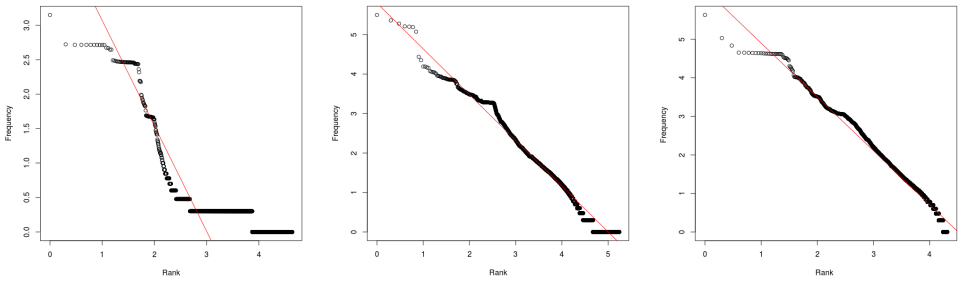
Como se puede observar los logs de todos los sitios analizados siguen una distribución tipo Zipf, esto es, la frecuencia de las peticiones de los objetos web de cada uno de ellos pueden ser modelados por una Zipf. Este resultado parece demostrar que la Zipf está dentro de la lógica que gobierna los sitios web, sobre todo si tenemos en cuenta la disparidad de sitios web: hemos incluido sitios muy demandados y poco demandados; sitios con muchos objetos web y con pocos; sitios con muchas peticiones y con relativamente pocas, y en todos aparece este comportamiento.



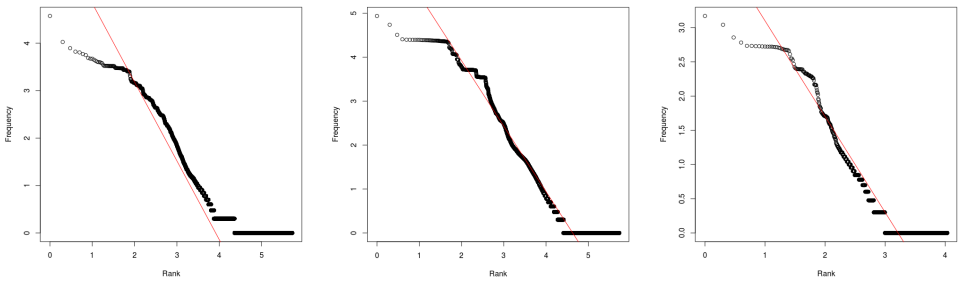
(a) rtve.es (b) bodasclick.com (c) eldiario.es



(d) ucm.es (e) otogami.com (f) ebuops.com



(g) grasia.fdi.ucm.es (h) wiki.rtve.int (i) fdi.ucm.es



(j) jira.rtve.int (k) pedro-munoz.es (l) cobisa.es

Figura 3.1: Representación de frecuencias tras aplicar logaritmos

Tabla 3.2: Datos obtenidos tras analizar las trazas

Sitio	Peticiones	Objetos únicos	α	x_{min}
rtve.es	130 millones	4.650.648	1,41	57
eldiario.es	8.961.959	1.287.733	1,16	5
bodaclick.com	13.987.028	1.337.045	1,30	33
ebuops.com	330.043	6.616	1,61	4
ucm.es	8.379.230	997.678	1,09	70
fdi.ucm.es	3.080.476	20.522	1,40	9
grasia.fdi.ucm.es	74.023	43.564	1,53	4
npcassoc.org	328.511	1.931	1,87	4
otogami.com	2.547.172	105.417	1,13	47
pedro-munoz.es	4.302.641	540.265	1,49	21
pdmmadridejos.es	221.062	53.610	1,53	2
cobisa.es	42.753	10.780	1,41	2
deportesmota.es	384.416	84.146	1,50	2
institut-francais-israel.org	168.753	7.749	1,63	2
jira.rtve.int	1.246.808	531.112	1,66	5
wiki.rtve.int	3.469.508	173.491	1,17	14

En la sección 2.3 se explicó que una Zipf se caracteriza por α, el exponente de la función que define la forma de ala distribución. En la bibliografía había un consenso basado en datos empíricos sobre la cota máxima del valor de α (Breslau et al., 1999; Krashakov et al., 2006) entorno a 1. Sin embargo, como se puede visualizar en los resultados, todas las trazas muestran que siguen una Zipf con $\alpha > 1$ y de hecho la mayoría de ellos presentan un valor mayor de 1,3. Esto implica que la distribución de probabilidad de los objetos web es mucho menos homogénea en las trazas de la tabla 3.1 que en las de la tabla de evidencias (tabla 2.3).

La menor homogeneidad de la probabilidad hace que el número de objetos necesario para obtener un cierto nivel de probabilidad acumulada, o un cierto volumen de frecuencia sea menor que en las evidencias pasadas. A priori esto hace que el uso de la frecuencia en cachés se plantee como una alternativa a tener en cuenta. Sobre este aspecto se hablará en el capítulo 4.

El hecho de que $\alpha > 1$ no ha sido destacado en la bibliografía. Desde nuestra perspectiva es un hecho relevante que debe ser revisado, sobre todo para entender sus consecuencias. Si se extrapola el resultado a otra capa, podría pensarse que existe una mayor predilección de los usuarios por un conjunto acotado de contenidos. Pero hay que tener cuidado para no errar

Tabla 3.3: Valor de Zipf por tecnología

Tecnología	Sitio	α
ASP.Net	fdi.ucm.es	1,40
Drupal	cobisa.es	1,41
Drupal	pedro-munoz.es	1,49
Drupal	pdmmadridejos.es	1,53
Drupal	grasia.fdi.ucm.es	1,53
HTML	eldiario.es	1,16
J2EE	otogami.com	1,13
PHP, J2EE	rtve.es	1,41
PHP	ucm.es	1,09
PHP	bodaclick.com	1,30
PHP	deportesmota.es	1,50
Tomcat	wiki.rtve.int	1,17
Struts2	ebuops.com	1,61
Tomcat	jira.rtve.int	1,66
Wordpress	institut-francais-israel.org	1,63
Wordpress	npcassoc.org	1,87

en el análisis. Lo relevante sería conocer qué tipo de objetos son los más demandados. Podría darse el caso de que el objeto más demandado de un site sea el logo, ya que está presente en todas las páginas. Por ello, para llegar a resultados en la vista es necesario un análisis más fino.

Visto que todas las trazas siguen una Zipf, se ha intentado hacer un análisis transversal, analizando las distintas columnas para intentar entender si existen relaciones implicitas. Por ejemplo, si analizamos la relación entre los objetos únicos y el número de peticiones, el rango varía desde el 0,58 % en npassoc.org hasta el 58 % que se da en grasia.fdi.ucm.es. No se aprecia ningún tipo de correlación entre ambas métricas, como tampoco la hay entre el porcentaje de objetos únicos y el exponente de la Zipf.

Si hacemos un análisis por tecnologías (ver tabla 3.3), no parece existir una relación global entre la tecnología y el valor de α. De hecho no existe correlación positiva ni negativa entre el valor de α y la tecnología. Esto no quita que el análisis de tecnologías nos lleve a encontrar algunos resultados interesantes: por ejemplo los sitios que están implementados con *Drupal* muestran un α entre 1,41 to 1,53. De hecho, si exceptuamos cobisa.es, el valor de α entre las trazas está entre 1,49 to 1,51. Esto nos podría llevar a pensar que quizá la implementación tecnológica, al ser un producto con su propia forma

Tabla 3.4: Valor de Zipf por sector

Sector	Sitio	α
Ayuntamiento	cobisa.es	1,41
Ayuntamiento	pedro-munoz.es	1,49
Ayuntamiento	pdmmadridejos.es	1,53
B2B y B2C	bodaclick.com	1,30
B2B deportes	ebuops.com	1,61
Educación	institut-francais-israel.org	1,63
Herramienta	wiki.rtve.int	1,17
Herramienta	jira.rtve.int	1,66
Investigación	grasia.fdi.ucm.es	1,53
Investigación	npcassoc.org	1,87
Juego	otogami.com	1,13
Noticias	eldiario.es	1,16
Poliderpotivo	deportesmota.es	1,50
Tele y Radio	rtve.es	1,41
Universidad	ucm.es	1,09
Universidad	fdi.ucm.es	1,40

de desarrollo, condiciona cómo es el consumo de las páginas y de los recursos.

El comportamiento anterior no ocurre con otras tecnologías: los servicios desarrollados con Java tienen un valor van del 1,13 al 1,66, o los basados en *PHP*, están en un rango que va desde 1,09 a 1,87, es decir, son muy dispares. Además de lo anterior, las cuatro trazas que presentan el menor valor de α están implementadas con cuatro tecnologías diferentes, algo similar a lo que ocurre con las trazas que tienen los mayores valores de α. Por estos detalles deducimos que no hay correlación entre la tecnología subyacente de un servicio web y el valor que muestra α para la distribución de frecuencia.

El análisis por sectores (tabla 3.4) nos permite extraes algunas conclusiones: debido a la naturaleza de los medios de comunicación, y al hecho de ser sitios muy conocidos, rtve.es y el diario.es presentan un enorme volumen de peticiones y objetos de negocio. Sin embargo, a pesar de pertenecer al mismo sector, los datos de los servicios son muy distintos, tanto en el ratio de objetos únicos por número de peticiones, como también por el valor que muestran de α o el valor de x_{min}.

Por su parte los ayuntamientos muestran un valor similar, pero es debido a que emplean la misma tecnología. Respecto a los sitios *B2B* y sitios que se dedican a vender productos, el valor de α es dispar y no presenta valores

similares, salvo el de ser mayor que 1.

Una última comparativa puede ser realizada comparando sitios de acceso libre frente sitios cuyo acceso es logado (jira.rtve.int y wiki.rtve.int). En el caso de los sitios de acceso basado en login, el valor de α varía desde el 1,17 hasta el 1,66, mientras que los sitios de acceso libre el valor encontrado va desde 1,09 hasta 1,87.

Para concluir, se podría decir que las tres mayores conclusiones que se obtienen de los resultados obtenidos al analizar las 16 trazas son:

1. La popularidad/frecuencia en los logs recolectados está mucho más concentrada en unos pocos objetos que en las evidencias anteriores. Este hecho podría ser empleado para diseñar políticas de cache más efectivas, ya que, para obtener un alto hit ratio bastaría con almacenar aquellos objetos que concentran una alta frecuencia.

2. Respecto a la predictibilidad, la fórmula 2.7 no contempla el hecho de que $\alpha > 1$. Es por tanto necesario determinar una forma de completar la ecuación.

3. Finalmente, tras analizar los resultados del análisis realizado sobre los logs, podemos concluir que en todos se da una Zipf, que tiene un $\alpha > 1$ y que el valor no depende ni del sector, ni de la tecnología: no importa el sector, el modelo de desarrollo o el modelo de ejecución, siempre se da una Zipf con un muy alta concentración de la probabilidad.

3.3. El caso de RTVE

Los trabajos de esta tesis se han centrado principalmente en los servicios que provee la Corporación RTVE dado el volumen de datos que presenta, la profundidad de datos de la que se dispone y del hecho que la distribución sigue una Zipf con $\alpha > 1$. En esta sección se va a bucear en estos datos, extrayendo información respecto de cómo actúan sus usuarios, de cómo actúan los robots o arañas que extraen contenido, cuál es la distribución de los tipos de contenidos y cómo son demandados o cómo evoluciona una Zipf en un día. Para este análisis se utiliza el log de un día de RTVE, aunque los resultados que se muestran se han comprobado con otros días obteniéndose comportamientos similares.

3.3.1. Evolución de una Zipf hora tras hora

Los estudios que aparecen en esta tesis (ver capítulo 2.3) se basan en el registro de las peticiones que hacen los usuarios a objetos de un sitio web. Dichas peticiones se agregan y analizan para obtener cuál es la distribución de las peticiones sobre dichos objetos, y utilizando los métodos mencionados en el capítulo 2.2.1, determinar cuál es el modelo que mejor caracteriza las distribuciones. Una vez encontrado del modelo se determina aquellos parámetros que los caracteriza, como por ejemplo, el valor de α de la muestra en caso de ser una Zipf.

En esta sección se va a realizar el análisis del log de RTVE por horas, con el propósito de ver cómo se distribuyen las peticiones a lo largo del día y finalmente si el modelo Zipf aparece en cada una de las franjas horarias. Para ello, el log de RTVE empleado en la sección 3.2 se ha separado en logs más pequeños, cada uno de ellos conteniendo las peticiones realizadas en cada una de las horas del día. Cada tramo horario se caracteriza en términos de usuarios únicos o número de peticiones, y se emplea para calcular la forma de la Zipf en cada una de las horas. Para más facilidad de lectura hemos dividido esta sección en dos subsecciones, la primera sobre la caracterización cuantitativa de la distribución, y la otra centrada en los aspectos relativos a las Zipf.

3.3.1.1. Análisis cuantitativo de la distribución de RTVE por horas

Tras descomponer el log de RTVE en tramos horarios y su correspondiente análisis, se obitene la imagen 3.2. Esta muestra cómo se distribuyen las peticiones que se reciben en los servidores de RTVE a lo largo del día. Se puede ver ella que el valle respecto de las peticiones que se reciben es a las 3 de la madrugada, donde el mínimo de peticiones es de 2.498.514, y que el máximo de la función de número de peticiones horarias se encuentra a las 20 horas de la tarde con un total de 8.202.378 peticiones.

La hora valle de peticiones coincide con la madrugada de España, para a lo largo del día ir estas aumentando hasta llegar al máximo de las 20 horas. El máximo se produce a la hora donde mucha parte de la población se prepara para la cena o esta cenando (alrededor de las 21 horas), mientras que el segundo máximo está localizado a la hora del almuerzo, esto es, a las 14 horas con 7.715.063 peticiones. Es llamativo que el número de peticiones baje a partir de las 21 horas, ya que es el prime time de televisión, esto es, la hora donde se emite el telediario, así como las series y contenidos más demandados. Para determinar el origen de esta anomalía habría que ver cuál

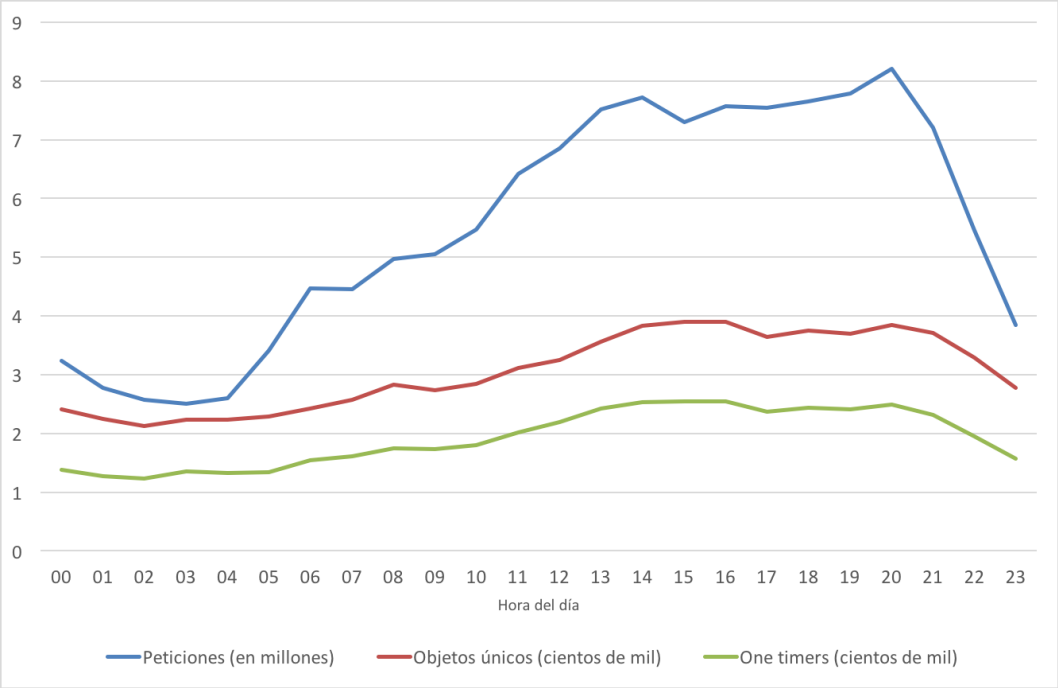


Figura 3.2: Peticiones, objetos únicos y one timers del site www.rtve.es

es la gráfica de usuarios activos por horas, cosa que se muestra en la imagen 3.3.

Al revisar la imagen se puede ver que el máximo de usuarios se obtiene a las 22 horas, es decir, una hora después del máximo de peticiones. La hipótesis que explica estas dos gráficas tiene que ver con los hábitos de consumo: desde las 18 horas en adelante los usuarios entran en el sitio y consumen contenido de forma libre, curioseando entre los mismos. Sin embargo, cuando empieza el prime time o el telediario los usuarios permanecen consumiendo el contenido, y por tanto, al no navegar para consumir contenidos distintos desciende el número de peticiones. Esto se manifiesta con la caída del número de peticiones desde las 21 que se puede ver en la figura 3.2 aunque el número de usuarios esté creciendo (figura 3.3). Esto mismo también aplica a la franja de las 14 a las 16, momento en el que está el telediario de la tarde (a las 15 horas).

Lo descrito anteriormente también aplica a los objetos únicos. En la figura 3.2 aparece la distribución de objetos únicos pedidos por hora, esto es, el número de objetos distintos que se pide en cada hora. Como se puede ver, sigue una tendencia similar al número de peticiones, aunque esta última distribución tiene cambios más abruptos. Si comparamos las dos distribuciones la conclusión es que a lo largo del día más usuarios entran en el servicio, y es-

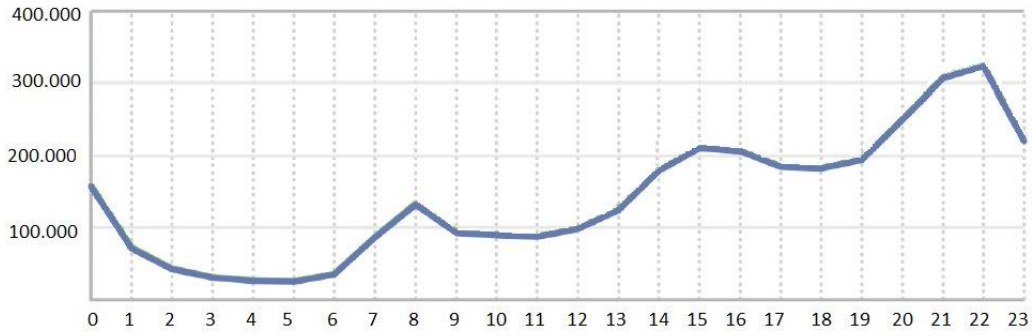


Figura 3.3: Usuarios del site www.rtve.es por horas

tos demandan cada vez más objetos diferentes. Teniendo en cuenta que cada página web implica un conjunto de objetos potencialmente distinto, el resultado es que el volumen de peticiones se dispara en un 400 % si comparamos la hora valle con la hora máximo.

Siguiendo con los objetos únicos, se puede observar que en las horas valles el número de peticiones promedio por objeto es de unas 10, mientras que en las horas pico dicha proporción es de casi 20 peticiones por objeto único. En la figura 3.4 se puede observar cómo a lo largo del día esta proporción aumenta hasta en un 100 %, aunque en la mayor parte del día la proporción está entre las 17 y las 22 peticiones por objeto. Más adelante veremos si este cambio afecta o no en la forma de la Zipf por hora.

En la figura 3.2 aparece una tercera distribución, los *one timers*. En la bibliografía (Breslau et al., 1999) se definen como sigue:

Definición 11. *Se define one timer como aquel objeto web que es accedido tan sólo una única vez en una muestra, o en otras palabras, aquellos objetos cuya frecuencia de accesos en la muestra es igual a 1.*

Cachear este tipo de objetos no ofrece ningún beneficio, ya que no van a volver a ser pedidos por los usuarios. Determinar que un objeto es un *one timer* es relevante a la hora de definir las políticas de caché de un sistema, ya que permite un ahorro de recursos al no ser considerados para ser almacenados temporalmente, con el ahorro en tiempo y recursos que conlleva.

La distribución de *one timers* del log de RTVE sigue el mismo patrón que la distribución del número de objetos únicos. De hecho las gráficas son equivalentes. De ello se deduce que a medida que hay más peticiones en el sistema, muchas de estas pasan a ser peticiones únicas provocadas por el libre acceso de los usuarios. Esto lleva a que cuanto más usuarios hay en el sistema, más peticiones habrá, y éstas serán a objetos que no se cachearán o

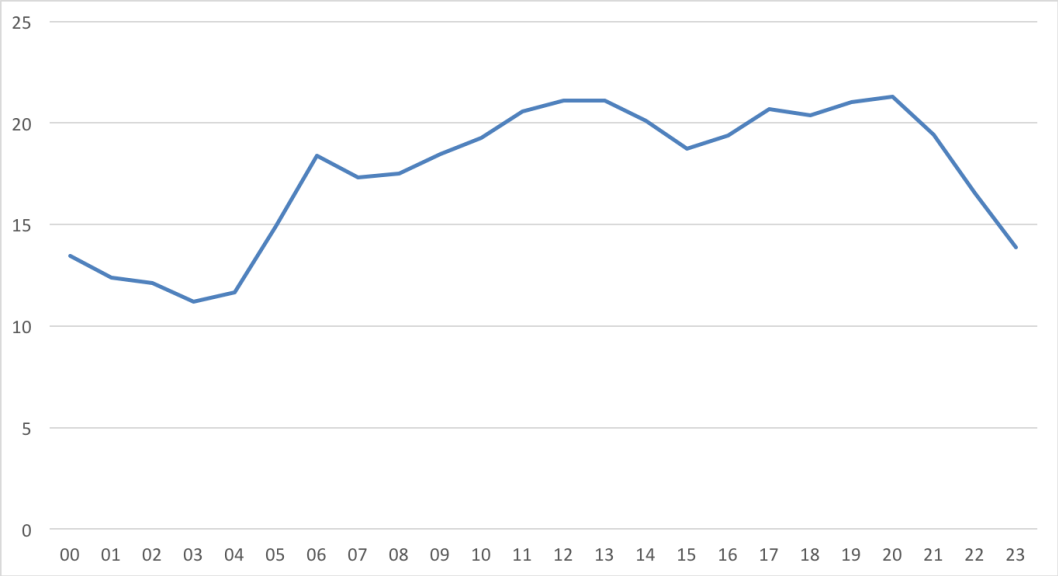


Figura 3.4: Proporción de peticiones por objetos en www.rtve.es

con frecuencias muy bajas.

Concluyendo esta subsección, el log de RTVE presenta una gran variabilidad de peticiones, objetos únicos, *one timers* y frecuencias de *one timers* a lo largo del día.

En la siguiente subsección se presenta la forma de Zipf por hora y se discutirá sobre la estabilidad de la Zipf en el día, así como si existen correlaciones entre los valores cuantitativos calculados en esta subsección y la Zipf por horas.

3.3.1.2. Valor de α por hora en la distribución de RTVE

Tras segmentar el log de RTVE tal y como se ha descrito en la subsección anterior, y siguiendo las técnicas descritas en la sección 3.2, se pretende determinar si la distribución Zipf modela la distribución de las peticiones en cada una de las franjas horarias, y si es el caso, el valor del exponente α para cada hora.

Utilizado las técnicas referidas, podemos concluir que en el caso de RTVE, al descomponer por horas el log, se obtienen distribuciones que siguen la forma de una Zipf. Respecto al valor de α, la figura 3.5 muestra su valor en cada uno de los segmentos.

Si miramos atentamente la figura veremos una línea con picos y valles y una línea horizontal. La línea horizontal está fijada al valor 1,41, que es el

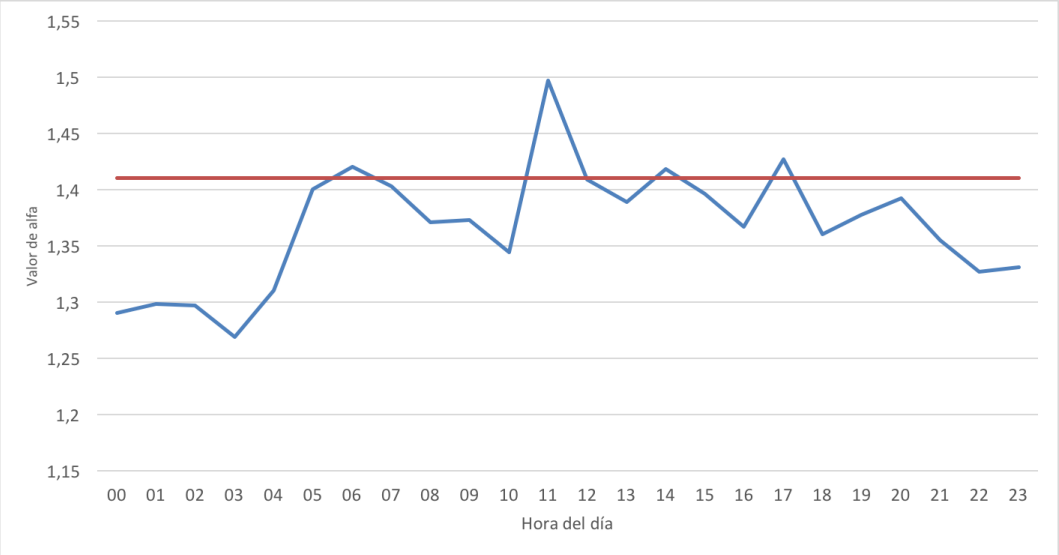


Figura 3.5: Variación del valor de alfa por hora en el sitio www.rtve.es

valor de α calculado en la sección 3.2. Como se puede observar el valor de α, línea que presenta valles y picos, va cambiando a lo largo del día, desde su valor menor 1,267 a las 3 de la mañana hasta el valor de 1,497 a las 11 de la mañana. La variación máxima entre el menor y el mayor valor es de un 18 %.

Si comparamos el valor por franja con el valor de 1,41 del día completo se aprecia que en la mayoría de los segmentos α tiene un valor menor que el valor diario, salvo varias excepciones: destaca el valor a las 11 de la mañana, donde α casi llega a un valor de 1,5.

La variabilidad del valor de α en una muestra puede tener un impacto en el rendimiento de una caché, y puede ser utilizado para mejorar el rendimiento de la misma. Para ello es necesario un modelo que permita determinar cuál es el rendimiento teórico de la caché en función del valor de α. Esto se discute en el capítulo 4, donde se introduce el concepto de predictibilidad, esto es, sobre la capacidad de poder determinar, por ejemplo, el hit ratio teórico de una muestra sabiendo que es una Zipf.

Utilizando las ecuaciones que se desarrollan en el capítulo 4, se ha calculado el hit ratio esperado en cada hora suponiendo que el número de objetos de la caché es de 1.000. Estos cálculos se pueden ver en la tabla 3.5, donde se puede ver cómo el hit ratio depende de forma directa del valor de α.

Para mostrar la casi perfecta correlación entre la variación de α y la variación del hit ratio así calculado, se va a utilizar el coeficiente de Pearson (Pita Fernández y Pértega Díaz, 1997). Dicho coeficiente varía entre 0 y 1, indicando el valor 0 que no hay correlación entre la variación de las distri-

Tabla 3.5: Hit ratio teórico por hora en la muestra de www.rtve.es

Hora	α	Objetos únicos	Hit ratio
00	1,29	240342	0,88
01	1,298	224268	0,89
02	1,297	212322	0,89
03	1,269	222912	0,86
04	1,31	222788	0,90
05	1,4	228764	0,95
06	1,42	242733	0,95
07	1,403	257048	0,95
08	1,371	283179	0,94
09	1,373	273453	0,94
10	1,344	284086	0,92
11	1,497	311916	0,98
12	1,409	324786	0,95
13	1,389	355721	0,94
14	1,418	383263	0,95
15	1,396	389657	0,94
16	1,367	390133	0,93
17	1,427	364792	0,96
18	1,36	375545	0,93
19	1,377	370284	0,94
20	1,392	385055	0,95
21	1,355	371089	0,93
22	1,327	329106	0,91
23	1,331	277063	0,91

buciones, mientras que el valor 1 implica que la correlación es perfecta. Para que el valor del coeficiente de Pearson sea válido, es decir, para considerar que la hipótesis tiene sentido, el valor de p-value debe ser menor de 0,05.

Al calcular el índice de correlación de Pearson se obtiene un valor de 0,96 con un valor de p-value igual a $6,661 * 10^{-15}$, lo que indica que la correlación es casi perfecta entre la variación de α y la variación del hit ratio.

Si atendemos a los valores obtenidos en la tabla 3.6, el conjunto de valores que tiene una mayor influencia en el valor de α resulta ser el porcentaje de one timers que tiene la muestra, y luego, el volumen total de peticiones de la misma.

Desde un punto de vista teórico, el hit ratio menor que se encuentra en

Tabla 3.6: Correlaciones entre distintas matrices y la matriz de valores de α

Conjunto de Valores	Índice de Pearson	p-value
Número de peticiones	0,613	0,0014
Número de objetos únicos	0,469	0,0204
Número de one timers	0,529	0,0078
Porcentaje de one timers	0,684	0,0001

la tabla 3.5 es de un 86 % con una caché de tan sólo 1000 objetos, mientras que el máximo es de casi un 98 %. Esos casi 12 puntos de diferencia suponen hasta un 12 % más de peticiones por hora que se resuelven en el servidor, lo que equivale entre 100 y 200 peticiones más por segundo. Como se puede ver, un mayor α implica una mayor capacidad de cacheo y por tanto una mayor eficacia en la estrategia de cacheo.

Otro detalle que se puede ver en la tabla es que para valores de α menores de 1,3 el hit ratio es menor al 90 %, y para valores mayores de 1,4 se supera el 96 %.

Hasta ahora se ha analizado la variabilidad del valor de α por horas, y la eficacia en el cacheo, pero no ha analizado qué factores cuantitativos de la muestra tienen mayor o menor influencia en la variación del exponente de la Zipf horaria. Para hacer este estudio se puede emplear modelos de correlación entre valores horarios: así se podría analizar la variación de, por ejemplo, el número de peticiones y cómo éste influye en la variación del valor de α obtenido en dicha hora.

En la subsección 3.3.1.1 se han presentado los valores del número de peticiones, número de objetos y *one timers* por hora. Para ver si existe alguna correlación entre estos valores y la variación de α en cada uno de los segmentos horarios se calcula el índice de Correlación de Pearson (Pita Fernández y Pértega Díaz, 1997).

La tabla 3.6 contiene los valores de dichas correlaciones, además de incluir un nuevo conjunto de valores que puede ser significativo: el porcentaje de *one timers* presentes en cada segmento horario.

Si atendemos a los valores obtenidos en la tabla 3.6, los cuatro conjuntos de valores, es decir, el número de peticiones, número de objetos únicos, número de *one timers* y el porcentaje de *one timers* influyen sobre la variación del valor de α por hora, tal y como refleja la columna *p-value*.

El coeficiente de correlación de Pearson se puede emplear para explicar parcialmente los valores de α de la siguiente manera: en el caso del porcentaje de *one timers*, el valor del coeficiente es de 0,684, esto significa que

$0{,}684^2 = 0{,}4624$ es la proporción de varianza compartida por la matriz de valores del porcentaje de *one timers* y la matriz de valores de α. Esto se puede interpretar como que el 46,2 % de la variación del valor de α depende del porcentaje de *one timers*. De la misma forma, para la muestra de RTVE, el número de peticiones de la muestra tiene un 37,5 % de influencia sobre la variación de α, el número de *one timers* influye en un 27 % y finalmente, el número de objetos únicos distintos es el valor que menos influye, compartiendo tan sólo un 22 % de varianza.

Atendiendo a los valores obtenidos en la tabla 3.6, el valor cuantitativo que más influye en la variación de la función Zipf es el porcentaje de *one timers*, por encima del número de peticiones o del número de objetos únicos.

Como conclusión, en esta subsección se puede ver que al descomponer la Zipf de RTVE por horas, cada segmento presenta una Zipf con valores de α que cambian en hasta un 18 %. Estas variaciones pueden provocar un cambio en el rendimiento de las cachés de hasta un 12 %. Estas variaciones pueden ser detectadas de forma automatizada para mejorar el tamaño y las políticas de las cachés. De hecho la variación en α, esto es, en la popularidad de los objetos, puede provocar un incremento indeseado de carga en los sistemas que soportan los servicos de RTVE.

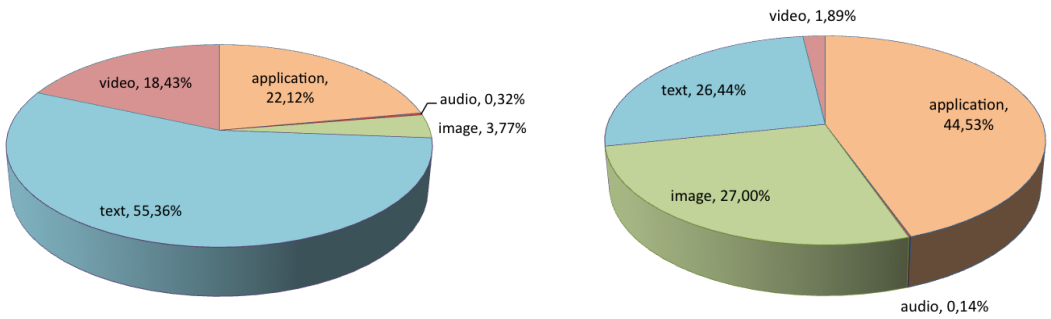
Además de lo anterior, se ha analizado qué valores cuantitativos tienen mayor influencia en la variación de la popularidad, siendo el porcentaje de *one timers* el valor más influyente. Podría ser de interés ahondar en cómo utilizar esta variación en la mejora de los sistemas, ya que es una métrica fácil de calcular.

En la siguiente subsección, y siguiendo con el log de RTVE se va a analizar la relación entre Zipf y el tipo de contenido.

3.3.2. Distribución de tipos de contenidos

Los servicios que ofrece RTVE incluye contenidos de todo tipo: audios, vídeos, noticias, encuestas, etc. Teniendo en cuenta el alto volumen de peticiones que presenta, y la gran variedad de contenidos, se ha considerado analizarlo para determinar si aparecen distribuciones Zipf al descomponer la Zipf de RTVE por tipo de contenido (ver sección 2.1.2). De este forma se pretende determinar si para el caso de RTVE, el descomponer su Zipf por tipo de contenido da como resultado distribuciones que también son modeladas por la Zipf.

Descomponer un log en los tipos de contenido no es una tarea sencilla, ya que el formato estándar de almacenamiento de logs (ver sección 2.2.1 no incluye el campo *content-type* (ver sección 2.1.2), y por tanto éste tiene que



(a) Basado en objetos únicos (b) Basado en peticiones

Figura 3.6: Porcentajes por tipo de contenido del site www.rtve.es

ser calculado a posteriori a partir de las peticiones de la muestra.

Hay dos formas diferentes de determinar el tipo de contenido de cada petición: realizar la petición almacenada de cada uno de los objetos web contra el servidor para así poder capturar la respuesta y leer la etiqueta content-type, o hacer una clasificación manual en base a patrones. Por ejemplo, todas las peticiones con extensión *png* son automáticamente consideradas como de tipo *image*.

En esta tesis se ha seguido el segundo modo de proceder ya que no supone incrementar la carga sobre del sistema productivo de RTVE. Para ello se ha procedido como sigue: primero se han obtenido los diferentes objetos web utilizando comandos de sistema (*awk*) para luego asociar a cada objeto web su tipo de contenido utilizando patrones o de forma manual. En caso de duda, se ha hecho la petición contra el servidor. Toda esta información se ha ido almacenando en una base de datos para poder hacer los cálculos posteriores.

El resultado puede verse en los gráficos que aparecen en la figura 3.6. En ella encontramos dos distribuciones: a la izquierda (figura 3.6a) se muestra los porcentajes de cada tipo de contenido de entre los objetos únicos servidos por RTVE. Así esta información muestra cómo se distribuyen los tipos de contenidos de entre los distintos objetos que son pedidos por los usuarios. Por su parte, la figura 3.6b muestra las proporciones de cada tipo de contenido que se encuentran dentro de las peticiones recibidas por el servidor. Ambas vistas son complementarias: la primera (figura 3.6a) muestra la proporción de tipos de contenidos de entre los objetos únicos, mientras que la segunda (figura 3.6b) lo hace de entre las peticiones que se reciben.

Respecto de los objetos únicos, tal y como se puede ver, la mayoría de

los objetos que se distribuyen son de tipo *text*, superando el 55 % de los objetos. El siguiente tipo más frecuente es el *application*, con el 22,12 %, y luego el *video* con el 18,3 % de los objetos. Los menos presentes son el *image* y el *audio*, suponiendo juntos el 4 % de los objetos. Es llamativo que los elementos multimedia sólo suponen un 23 % del total de los objetos, lo cuál supone que la mayoría de los elementos que se utilizan en RTVE para sus servicios son elementos textuales (noticias, páginas web, etc.) y elementos de tipo *application* (css, *JSON*, etc.).

En la figura 3.6b se puede ver la distribución de tipo de contenidos de entre las peticiones que se reciben. Llama la atención que el tipo más demandado sea el *application*, 44,53 %, muy por delante del tipo *image*, 27 %, y del *text*, 26,44 %. El resto de tipos (*audio* y *video*) sólo recibe el 2 % de las peticiones.

Como se puede, ver hay una gran diferencia entre los tipos de los objetos que existen en el servidor y la distribución de sus frecuencias: el resultado más impactante es que el tipo *image* tiene una importancia 9 veces mayor entre las peticiones que entre los objetos, mientras que el *application* duplica su proporción. El caso del *text* pasa del 55 % al 26 %. Este cambio en las proporciones tiene que ver con los modelos de consumo de la actualidad, donde casi todas las páginas son dinámicas, donde la mayoría del contenido dinámico se genera a partir de *APIS* con datos en formato *JSON*, y donde se suele provocar la carga dinámica de imágenes. De hecho, el tipo de contenidos más demandado es *application/json* que representa el 37 % del total de peticiones del log de RTVE.

Además de lo anterior, y con la irrupción de la movilidad, el paradigma de desarrollo nativo dinámico implica el intercambio de peticiones basadas principalmente en *JSON*, así como la carga de muchas imágenes. No es por tanto casual que estos tipos de contenidos sean los más demandados dentro de los servicios de RTVE.

Aprovechando que se ha separado el log, las distribuciones de cada tipo de contenido han sido analizadas para determinar si presentan una Zipf. En la tabla 3.7 se muestra el valor de α calculado para cada una de las distribuciones.

Analizando la tabla 3.7 se puede ver que al descomponer el log de RTVE, que seguía una Zipf, en tantas distribuciones como tipos de contenidos, se obtiene que todas estas nuevas distribuciones también siguen una Zipf, cada una con un valor de α propio.

En el caso del tipo *application*, se puede ver que su valor de α es próximo al de la distribución completa (1,41). Esto es lógico, ya que el tipo *application* supone casi la mitad de las peticiones, y por tanto es el tipo que ejerce una mayor influencia sobre la distribución de peticiones del log.

Tabla 3.7: Valor de α para cada tipo de contenido del site www.rtve.es

Tipo contenido	α
application	1,36
audio	1.08
image	1.14
text	1.22
video	0.96

Respecto del tipo *text*, que en el caso de RTVE son en su mayoría páginas web, se puede ver que el valor de α es menor de los 1,41. Este resultado está en consonancia con los encontrados en el capítulo 5 donde se analiza la capa de vista. En él se comenta que los valores de α encontrados en el servidor son mayores que en la capa de cliente, al igual que pasa con el tipo de contenido *text*.

Finalmente, respecto de los tipos de contenido multimedia, el valor de α es inferior, llegando en el caso de *video* a ser menor que 1. El caso de las imágenes, el valor de α puede estar influido por el conocido efecto *Tricle Down* (Doyle et al., 2002), ya que las imágenes suelen estar cacheadas por los navegadores, de manera que el valor de α calculado es menor del que resultaría si no hubiese caché en los navegadores.

Teniendo en cuenta que los objetos multimedia suelen ser estáticos, mientras que los de *application* o los *text* pueden ser dinámicos, desde el punto de vista de la coherencia de las cachés se impone una política diferenciada: los objetos estáticos que suelen tener más tamaño y que no cambian pueden ser almacenados con un alto tiempo de expiración. En el lado opuesto están los objetos dinámicos que tienen una tasa de cambio alta: estos deben ser almacenados con un tiempo de expiración corto, y además, si tienen una muy alta frecuencia de acceso, sería prudente tener mecanismos de refresco del contenido para mantenerlos en las cachés.

La conclusión que se obtiene de esta sección es que al dividir una Zipf como la de RTVE en función del tipo de contenido, se obtienen tantas Zipf como tipos de contenido. Además de lo anterior, las proporciones del tipo de contenido es muy diferente si la medimos dentro de la distribución de objetos únicos o si la medimos dentro de la distribución de peticiones del servidor.

A continuación se va a analizar cómo los robots, arañas y otros mecanismos automatizados de recogida de información en base a peticiones HTTP afectan a la distribución de contenidos de RTVE, así como si estas peticiones también siguen una Zipf.

3.3.3. Impacto de los robots y arañas

Los servicios web, la mayoría del tiempo, atienden a peticiones que vienen de usuarios "legítimos"que consumen los contenidos que se ofertan. Pero en la actualidad también se puede encontrar la presencia de robots o arañas que actúan como usuarios con diferentes propósitos: algunos buenos, como por ejemplo que los buscadores más utilizados, como Google o Yahoo, lean las páginas de un sitio para indexarlas a la hora de realizar búsquedas, y otros malos, que utilizan las páginas de los sitios web para replicarlas u otros propósitos negativos. Estos robots analizan las páginas en busca de más enlaces y de información que será utilizada por los motores de búsqueda.

Un robot se identifica utilizando un campo del protocolo HTTP, el *user-agent*, que se emplea para identificar al usuario (Fielding et al., 1999). Dicho campo es enviado en la petición de una página y sirve para determinar qué tipo de usuario es el que está realizando la petición. Por ejemplo, la araña de Google envía el campo *user-agent* de la cabecera de petición con el valor *Mozilla/5.0 (compatible;Googlebot/2.1; +http://www.google.com/bot.html)*. Así, desde el servidor se puede actuar, permitiendo o denegando el acceso al recurso que solicita.

El campo *user-agent* es registrado en los logs, lo que permite analizar el impacto de los robots en los servicios web. En esta tesis el log de RTVE se va a analizar para determinar dicho impacto, así como para determinar si las peticiones que hacen los robots se pueden modelar utilizando la distribución Zipf.

Para realizar dicho análisis se va a seguir el siguiente modo de proceder:

1. El primer paso es determinar qué peticiones han sido realizadas por robots. La mayoría de servicios identifican que se trata de un robot incluyendo la cadena *bot* en el *user-agent*, como el ejemplo anterior de Google. Por tanto, el primer filtrado consistirá en seleccionar aquellas peticiones donde el *user-agent* contenga la cadena *bot*.

2. Además de las peticiones obtenidas con la regla anterior, hay otra serie de posibles robots fácilmente detectables: estos tienen como *user-agent* literales como *java*, *nodejs* u otros nombres de tecnologías. Los *user-agent* anteriores son los que ponen por defecto las distintas tecnologías cuando se hacen programas para leer páginas y procesarlas. Por tanto, las peticiones que cumplen con esta regla también son seleccionadas.

3. Finalmente se hace un repaso mediante catas aleatorias y se completó el listado de peticiones provenientes de *bots*.

Tabla 3.8: Datos obtenidos en las peticiones de robots al site www.rtve.es

Descripción	Valor
Número de peticiones hechas por robots	920.468
Número de objetos únicos accedidos por robots	479.357
Número de one timers	376.553
Porcentaje de peticiones hechas a objetos de tipo *text*	97 %
Porcentaje de peticiones sobre el total	0,69 %

Siguiendo los pasos descritos anteriormente se genera un log con todas las peticiones realizadas por robots. Éste es analizado obteniendose los datos que aparecen en la tabla 3.8.

Como se puede ver en la tabla 3.8, el número total de peticiones que se realiza por robots supone un 0,69 % del tráfico que se llega a los servidores de RTVE. Si tenemos en cuenta que los robots no suelen llamar a los objetos referenciados en una página, y que el promedio de elementos que incluye una página de RTVE es de 100, el tráfico equivalente a ese número de peticiones que hubiese generado un usuario "legítimo"sería de unos 9.000.000 de peticiones, casi el 7 % del tráfico del portal, una cifra nada desdeñable de peticiones que no se llegan a producir.

Los usuarios de RTVE en el log analizado generaron 10.145.337 peticiones a páginas vistas en un día, con lo que los robots fueron responsables del 9,07 % de las peticiones a páginas en el portal www.rtve.es. Como se puede ver la actividad de los robots es relevante en sitios que posicionan sus contenidos como el que se está estudiando. De hecho, cada segundo se producen 10 peticiones desde robots a los servidores de RTVE.

Aplicando las técnicas descritas en este capítulo se calcula si las peticiones que realizan los robots se pueden modelar como una distribución Zipf. El resultado es afirmativo y se calcula cuál es el valor de α, obteniéndose un valor de 0,582. En la figura 3.7 se puede apreciar cómo tras aplicar logaritmo al ranking y a la distribución de las peticiones de los robots, aparece la forma característica de una Zipf.

Finalmente para terminar esta subsección se han extraído algunos datos sobre los distintos robots que acceden a RTVE. En concreto, se ha calculado cuántos robots distintos realizan peticiones a RTVE, y si la frecuencia de peticiones de cada robot sigue o no una Zipf. En la figura 3.8 se muestra la forma típica de una Zipf tras aplicar logaritmo tanto al ranking de los robots en el eje X, como a la frecuencia de cada uno de ellos en el eje Y. Al calcular el valor del exponente de la Zipf se encuentra que α vale 3,02. Este valor excede en mucho el mayor valor encontrado para distribuciones de peticiones

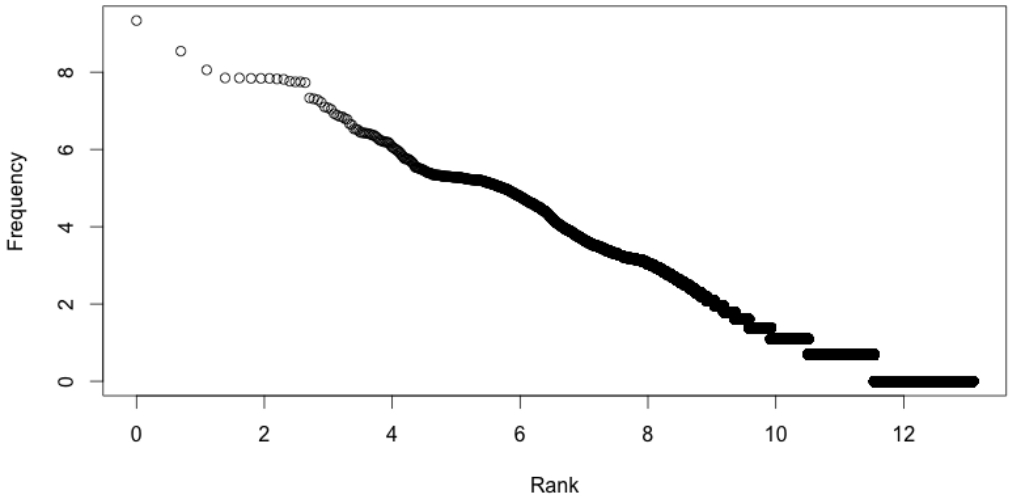


Figura 3.7: Figura de frecuencias de peticiones de robots

en el dominio de la web.

Respecto a datos cuantitativos de robots, el número de ellos que se han encontrado es de 527, de los cuales la 256 sólo hacen una petición (*one-timers*). En la tabla 3.9 se enumeran aquellos robots que han realizado más de 10.000 peticiones en un día.

Como se puede ver en la tabla, la mayoría de los robots que aparecen son arañas de los buscadores más extendidos: el robot de Google es el que hace más peticiones al servidor con un promedio superior a las 3 peticiones por segundo para posicionar los contenidos de RTVE. Bing por su parte realiza un promedio similar al de Google. A continuación encontramos más buscadores como MSN o Yandex. Pero no todos los robots son de buscadores: por ejemplo el noveno robot que hace más peticiones resulta ser la web archive.org que mantiene copias de las páginas de sitios con fines de preservación. Este último robot hace, en promedio, una petición cada ocho segundos.

Para concluir esta subsección, al analizar el impacto de los robots en el sito de RTVE se ha podido comprobar que son responsable de más del 9 % de las peticiones que se realizan a páginas web. Además de lo anterior, se ha podido validar que Zipf modela la distribución de las peticiones que los robots hacen a dichas páginas, mostrando un valor de α igual a 0,582.

Respecto a los robots, hay al menos 527 robots que han accedido a RTVE para descargar contenido. Al analizar si la distribución de peticiones que ha

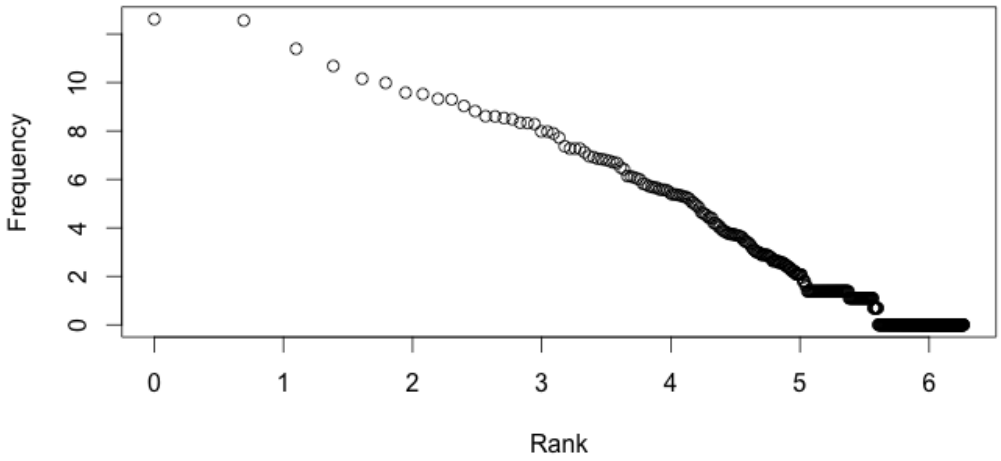


Figura 3.8: Figura de frecuencias de robots que acceden a www.rtve.es

Tabla 3.9: Robots que han hecho más de 10.000 peticiones en un día al site www.rtve.es

Robot	Número de peticiones
http://www.google.com/bot.html	300.653
http://www.bing.com/bingbot.htm	284.498
http://www.ahrefs.com/robot/	88.556
http://search.msn.com/msnboot.htm	69.204
Twitterbot	21.656
http://yandex.com/bots	14.493
http://superfeedr.com	13.681
http://commoncrawl.org/faq/	11.202
http://archive.org/details/archive.org_bot	10.965

hecho cada robot sigue una Zipf, hemos descubierto que sí, que la Zipf está presente pero con un valor altísimo de α igual a 3,02.

3.4. Conclusiones

En este capítulo se ha abordado el análisis del log de servidor de dieciséis sitios web, todos ellos obtenidos en 2014, con el propósito de medir de forma empírica si siguen una distribución Zipf, y en su caso el valor de alpha. Los resultados que se han obtenido apuntan a la presencia de la distribución Zipf en todos los sitios, con un valor de α mayor que en las muestras empíricas que se pueden encontrar en la bibliografía. Esto implica un mayor sesgo en la distribución de las frecuencias de los objetos web en los sitios, y una mayor importancia de la popularidad a la hora de implementar soluciones, como por ejemplo a la hora de hacer cachés, para los sitios analizados.

Aparte del análisis del valor de α, en este capítulo también se ha hecho una comparativa entre los distintos sitios utilizando dos características: el sector al que pertenecen y la tecnología con la que se han desarrollado. La conclusión a la que se llega, es que tanto el sesgo tan pronunciado en las distribuciones como el hecho de ser una Zipf no depende de dichas variables, ya que no existe una correlación entre ellas y el valor de α que se obtiene.

El sitio web de RTVE ha sido objeto de especial análisis en el capítulo debido al gran volumen de peticiones que presenta, a la gran diversidad de tipos de contenidos que sirve y finalmente a que presenta una Zipf con un valor de α con mucho sesgo. El análisis que se ha realizado descompone el log de un día por franjas horarias, por tipo de contenido y finalmente analiza la relevancia de los robots.

Cuando se descompone el log de RTVE que sigue una Zipf por horas, el resultado son 24 logs que también siguen una distribución basada en Zipf. Esto es, al descomponer el log de rtve.es por hora encontramos que todos los subconjuntos también siguen una Zipf, donde el valor de α varía en hasta un 18 %.

Del análisis por hora, se desprende un resultado novedoso: al menos en el site rtve.es, el porcentaje de *one timers* presentes en la muestra es la métrica que mejor predice la cuantía de variación de la popularidad de la misma. Puede ser de interés analizar el uso de esta métrica para mejorar el rendimiento de sistemas, cosa que nadie ha estudiado.

Un último resultado interesante ha emergido del análisis por hora: el máximo de usuarios únicos en la web se da antes que en la televisión. De hecho, cuando empieza el prime time el usuario permanece en el sitio consumiendo

el contenido, mientas que antes de esa franja horaria, los usuarios tienden a navegar y consumir de forma más aleatoria. Es como si el usuario antes del prime time navega y consume de forma libre, para después concentrarse en los contenidos más relevantes del día.

La descomposición de un log por otros criterios, como es por el tipo de contenido, también da como resultado distribuciones basadas en Zipf para todos los subconjuntos. Cada tipo de contenido muestra un α diferente así como proporciones distintas cuando se comparan magnitudes como el número de peticiones por tipo de contenido frente al número de objetos únicos por tipo de contenido. En el caso de RTVE se da una mayor relevancia del tipo de contenido *application* por la irrupción del consumo móvil y de teles conectadas.

Esta descomposición lleva a una cuestión que puede ser relevante: vista la distribución de los distintos tipos de contenidos, cabe estudiar si las políticas de cachés deberían ser diferentes para cada tipo de contenido. En cualquier caso, en lo que respecta a la coherencia, los distintos tipos de contenidos implican diferentes políticas de actualización, ya que los contenidos multimedia, por ejemplo, no requieren de actualización, cosa que si ocurre con los de tipo *application* o *text*.

Finalmente, en el capítulo se ha analizado el papel de los robots en el log de rtve.es. En el análisis se muestra que la distribución de los objetos accedidos por los robots, así como la frecuencia de peticiones que hace cada robot siguen distribuciones Zipf. Además del resultado anterior, centrado en las Zipf, los robots fueron responsables del 9 % de las peticiones a páginas web del portal, cifra nada desdeñable. Este volumen se justifica por la dependencia de los sitios web de los buscadores: un buscador permite acercar el contenido a los usuarios, para lo cual, los buscadores emplean robots que leen las páginas y las indexan en base a los términos que en ellas aparecen.

En el próximo capítulo se van a estudiar los dos aspectos primeros, esto es, la predictibilidad y el impacto en las políticas de caché de la alta concentración de la frecuencia.

Capítulo 4

Benchmark para evaluación del impacto en cachés del parámetro α

En el capítulo anterior se presentaron 16 sitios web de los cuáles se obtuvieron los logs de servidor. Tras un análisis de los mismos se llegó a a la conclusión de que todos los sitios presentan un valor de $\alpha > 1$. En este capítulo se van a estudiar las consecuencias de este hecho desde dos perspectivas distintas: por un lado se va a analizar el impacto en la capacidad de calcular el valor del hit ratio teórico que se podría obtener para la muestra, y por el otro cómo impacta en las políticas de caché.

4.1. Predictibilidad del hit ratio para distribuciones Zipf

Se entiende por predictibilidad del hit ratio, la capacidad de determinar el hit ratio máximo que una muestra podría alcanzar. Este valor es teórico ya que el hit ratio real depende de la política de caché y de cómo es el tráfico de la muestra.

El hecho de que una distribución siga una Zipf, y tal como se describe en la sección 2.3, permite estimar el hit ratio (Shi et al., 2005) utilizando como variables α, k, el tamaño de la caché, y N, el número de objetos únicos utilizando la fórmula 2.7. Ésta sólo contempla que $\alpha \leq 1$. Por ello, en esta sección se va a completar la fórmula para contemplar el caso que falta, el de α mayor que 1.

De acuerdo a Shi et al. (2005), se define el número armónico de orden n

sobre α como sigue:

$$H_\alpha(n) = \frac{1}{1} + \frac{1}{2^\alpha} + ... + \frac{1}{i^\alpha} + ... + \frac{1}{n^\alpha} = \sum_{i=1}^{n} \frac{1}{i^\alpha} \tag{4.1}$$

La función así definida es una función hiper geométrica que es convergente o divergente dependiendo del exponente. El cálculo de $H_\alpha(n)$, para un número k dado es sencillo. Pero en el caso de que $n \to \infty$ la cosa cambia. Shi et al. (2005) trata este caso cuando $\alpha < 1$, donde se resuelve utilizando una aproximación basada en la integral.

En el caso de que $\alpha > 1$, el número armónico converge para todo k, y de nuevo se calcula fácilmente para un valor dado. Pero en el caso de nuevo de que $n \to \infty$, al ser una serie divergente, el número armónico puede ser aproximado a la *función Zeta de Riemann* evaluada en α: $\zeta(\alpha)$.

Para relacionar el armónico con el hit ratio, Shi et al. (2005) concluyeron que el valor acumulativo de probabilidad para el k_{esimo} objeto web más popular en una Zipf es igual al valor estimado del hit ratio para una caché de tamaño k. Para calcular el valor acumulado de probabilidad, basta con dividir el armónico en k y α divido por el armónico sobre α para N, el número de objetos diferentes de la muestra. La fórmula del hit ratio quedaría así:

$$h = \Phi(k, \alpha) = \frac{H_\alpha(k)}{H_\alpha(N)} \tag{4.2}$$

Si consideramos que la muestra puede llegar a tener un valor infinito de objetos, entonces podríamos aproximar la fórmula a:

$$h = \Phi(k, \alpha) = \frac{H_\alpha(k)}{H_\alpha(N)} \approx \frac{H_\alpha(k)}{\zeta(\alpha)} \tag{4.3}$$

Usando las fórmulas 2.7 y 4.3, la fórmula general que permite calcular el hit ratio teórico de una Zipf es:

$$h = (\frac{k}{N})^{1-\alpha} \quad \text{si } \alpha < 1, \quad \frac{log(k)}{log(N)} \quad \text{si } \alpha = 1, \quad \frac{H_\alpha(k)}{\zeta(\alpha)} \quad \text{si } \alpha > 1 \tag{4.4}$$

Para visualizar el significado de la fórmula 4.4, una simulación de cómo varía el hit ratio en función de α se muestra en la figura 4.1. En ella se ha hecho una simulación fijando el tamaño de la caché (k) igual a 10.000, y el número de objetos únicos (N) a 4.000.000.

Como se puede ver, el resultado de la simulación muestra algunas de las conclusiones que se apuntaron: cuanto mayor es el valor de α, mayor es el hit

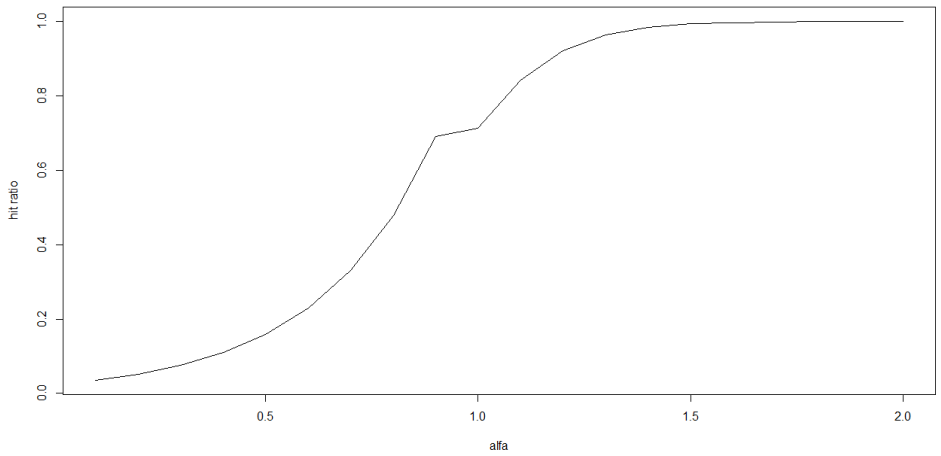


Figura 4.1: Hit ratio estimado para una caché de tamaño 10000

ratio teórico. Como se ha apuntado otras veces, este resultado era más que esperado, ya que la mayor concentración de frecuencia en pocos objetos hace que la eficiencia de una caché aumente.

También se puede observar que cuando en una muestra α tiende a ser mayor que 1, la eficiencia de una posible caché aumenta de forma muy rápida, y un alta hit rate se puede conseguir incluso con una cache pequeña. Por lo tanto, una cache con alto rendimiento podría implementarse bajo estas condiciones. Incluso sería interesante determinar cómo se comportan las políticas de caché bajo estas condiciones.

Estas cuestiones se van a analizar en la siguiente sección.

4.2. Rendimiento de cachés bajo distribuciones Zipf con $\alpha > 1$

El objeto de esta sección es determinar cómo se comportan, en relación con políticas de caché, distribuciones que siguen una Zipf con $\alpha > 1$. Para tal fin se han recogido 4 trazas reales (ver tabla 4.1) como ficheros tipo access.log, obtenidas desde los nodos más cercano al usuario. Algunos de los servicios emplean múltiples nodos, así que se han unidos los diversos logs para simular la carga que recibió el servicio, pero como si fuese servida por un único nodo. De esta forma podemos simular el comportamiento global de una caché. Lógicamente, al haber varios nodos involucrados, el efecto de la caché se diluye.

Dada una distribución de frecuencias que sigue una Zipf, si una cache se dimensiona con el número de objetos a almacenar entonces las políticas basadas en la popularidad serán las que den los mejores resultados. Hay que tener en cuenta que una Zipf se basa en una preferencia por unos pocos objetos frente a los demás, y mantener los más demandados en la caché siempre reportará un alto hit ratio (ver punto anterior).

Sin embargo, el número de objetos no es un método común en implementaciones de la industria (Fog Heen et al., 2013; Squid, 2015; Team, 2013), ya que es más útil configurar una caché restringiendo el tamaño que puede ocupar en MB de objetos almacenados. En ese caso, dos factores se convierten en cruciales a la hora de elegir la caché: la frecuencia y el tamaño de los objetos. Por ello cobra especial importancia si los mecanismos de cachés utilizados en la bibliografía y en las soluciones siguen siendo igual de efectivas.

Por último, es interesante validar si una distribución basada en Zipf puede tener algo hit ratio en caches pequeñas. Hay que tener en cuenta que el hit ratio está relacionado con la frecuencia de los objetos, así como con el patrón de generación de las peticiones, esto es, en qué momento los objetos empieza a ser pedido.

A continuación se va a comentar qué políticas se van a emplear, cuál es el conjunto de datos, la metodología y finalmente los resultados. Los datos de este capítulo han sido sacado de Zotano et al. (2015a).

4.2.1. Políticas de caches seleccionadas

Para entender cómo se comportarán distribuciones de peticiones que siguen una Zipf con valor de α mayor que 1, en simulaciones que involucran cachés con distintas políticas de caché, se ha hecho una selección de aquellas políticas empleadas en soluciones comerciales y otras que han sido de interés en la literatura. Para más información sobre las caches ver sección 2.1.

En el estudio se han empleado las políticas GDSF y LRU (Wong, 2006) que se emplean en soluciones comerciales (Squid, 2015; Team, 2013; Fog Heen et al., 2013). Además se han seleccionado tres políticas ampliamente usadas en estudios anteriores: la política de caché GDS, la LFU y la LFU-DA (Wong, 2006).

Finalmente, se introduce un último algoritmo, Perfect GDSF (GDSF*), el cual realiza una mejora sencilla respecto al GDSF. En vez de utilizar la frecuencia únicamente de los objetos que están en la caché, el GDSF* mantiene un listado de las frecuencias de todos los objetos, tanto si están en la caché como si no. De esta forma los objetos mantienen su historia de peticiones, cosa que en principio debería ayudar a mejorar el hit ratio.

Tabla 4.1: Logs simulación de caché

Sitio	Objetos Únicos	Peticiones	α	Tamaño Completo (MB)	Tamaño Medio (KB)	Espacio trabajo (MB)
rtve.es	649.145	15.992.330	1,37	121.172	5,51	3.497
eldiario.es	1.272.517	7.307.473	1,13	78.047	2,94	3.651
bodaclick.com	1.050.570	10.000.000	1,13	117.625	37,71	38.689
ucm.es	208.051	2.736.795	1,09	91.003	30,12	8.240

Aunque hay más políticas, los factores de caches más habituales están representados, permitiendo las elegidas hacer comparativas retrospectivas con resultados más antiguos.

A continuación se presenta la colección de datos que se han utilizado para la simulación.

4.2.2. Banco de datos para la simulación de caches y preparación de las pruebas

Para simular el cacheado de distribuciones que siguen una Zipf con un valor de α mayor que uno, se ha seleccionado cuatro sitios web que cumplen con dicha condición, y se ha obtenido el log de cada uno de ellos. Cada log está formateado en base al formato CLF (ver sección 2.2.1) y para cada uno de ellos se han ordenado las peticiones que contienen de forma cronológica. De esta forma, el banco de pruebas consiste en un registro cronológico de las peticiones reales que se hicieron para cada uno de los cuatros sitios seleccionados.

La tabla 4.1 muestra las trazas obtenidas: la tabla contiene información del sitio web, el número de objetos únicos del log, el número de peticiones, el valor de α, el tamaño total contenido distribuido en MB, el tamaño medio de un objeto y finalmente el espacio de trabajo. Como se puede apreciar todas las trazas cumplen las condiciones mencionadas.

Prestamos especial atención al sitio rtve.es, el sitio de RTVE por ser un sitio muy demandado y que presenta un valor muy alto de α, 1,37. También aparece eldiario.es, otro sitio de noticias que va a permitir hacer una comparativa de sector. Finalmente se utilizan bodaclick.com y ucm.es, los cuáles cumplen con las especificaciones y tienen un volumen alto de peticiones.

Como se ha comentado en la sección anterior, se van a emplear las políticas de caché LRU, LFU, LFU-DA, GDS, GDSF y Perfect GDSF (Wong, 2006). Para poder hacer la simulación se ha hecho un conjunto de classes Java que

Tabla 4.2: Tamaños de caché usados en MB

Sitio	0,1 %	0,2 %	0,3 %	0,4 %	0,5 %	0,6 %
rtve.es	3,497	6,994	10,491	13,988	17,485	20,982
eldiario.es	3,651	7,302	10,953	14,604	18,255	21,906
bodaclick.com	38,689	77,378	116,067	154,756	193,445	232,134
ucm.es	8,24	16,48	24,72	32,96	41,2	49,44

implementan cada una de las políticas de caché. El programa va leyendo línea a línea los logs y para cada una de ellas realiza la correspondiente petición de entrada en la caché. Una vez terminado de procesar todas las líneas del log, un fichero con los datos de la simulación, así como el hit ratio conseguido se escribe.

Para simular la implementación de cada una de las caches, se ha recurrido a la utilización de una base de datos *h2* embebida, de tal manera que las operaciones de inserción, actualización, actualización del número de accesos, borrado de un objeto y reordenación de objetos han sido traducido a sentencias SQL. Haciendo más sencillo el desarrollo y más rápido la ejecución de la misma.

La implementación ha utilizado dos tablas distintas para cada políticas: una para la contabilización de las peticiones, manteniendo para cada objeto el número de veces que ha sido llamado (su frecuencia momentánea), y una segunda tabla que almacenaba la información de la caché, los objetos dentro de la misma, los factores de envejecimiento, etc.

Hay una asunción que comparten todas las implementaciones de cache en la simulación: los objetos de una caché siempre son válidos y no caducan. Esta asunción es habitual en simulaciones como la que se ha realizado.

Finalmente, para completar la preparación de la prueba hay que seleccionar el tamaño de la caché. En vez de seleccionar un valor arbitrario, lo que se va a hacer es elegir como tamaño de cache 6 valores: desde 0,1 % al 0,6 % del tamaño del espacio de trabajo de la muestra. De esta forma se puede medir el efecto de utilizar caches pequeñas en distribuciones como las de la tabla 4.1, así como evoluciones el hit ratio a medida que vamos aumentando el tamaño de caché. Los tamaños aparecen en la tabla 4.2

Como se puede ver, y si tomamos como ejemplo rtve.es, el tamaño ocupado por el espacio de trabajo (tamaño de todos los objetos a distribuir) es de 3,5 *GB*. Para las caches, el máximo tamaño de caché que se va a emplear en las pruebas corresponde a tan sólo $21MB$, el 0,6 % del espacio de trabajo.

A continuación se muestra y comentan los resultados del test.

4.2.3. Resultados de las simulaciones

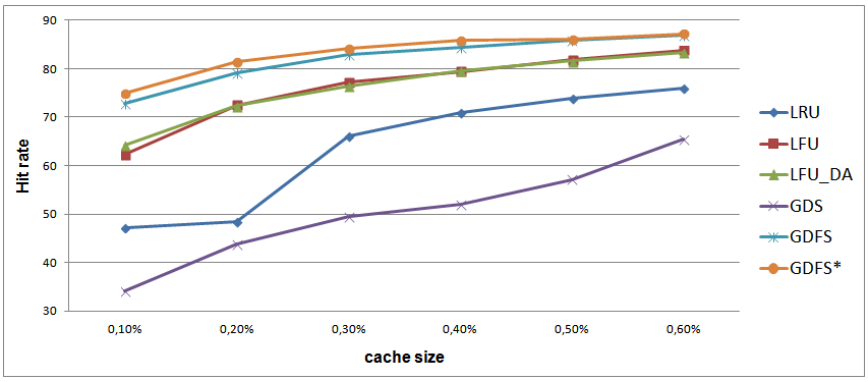
Siguiendo las condiciones mencionadas anteriormente se ha realizado la simulación de las trazas que aparecen en la tabla 4.1 con los tamaños de caché que aparecen en la tabla 4.2 utilizando un desarrollo propio que implementa cada una de las políticas de caché a validar: LRU, Perfect LFU, LFU DA, GDS, GDSF and Perfect GDSF. En la figura 4.2 se puede ver cuál es el valor del hit ratio que se ha obtenido para cada traza, con cada una de las políticas de caché y con cada tamaño de caché.

Como puede ver en la imagen, y como era de esperar, los cuatro resultados, muestran un comportamiento similar: el hit ratio crece con el tamaño de la caché, lo cual es lógico, ya que al haber más sitio disponible más objetos pueden permanecer en la cache, y por tanto hay mayor probabilidad de encontrar un objeto pedido anteriormente en la cache.

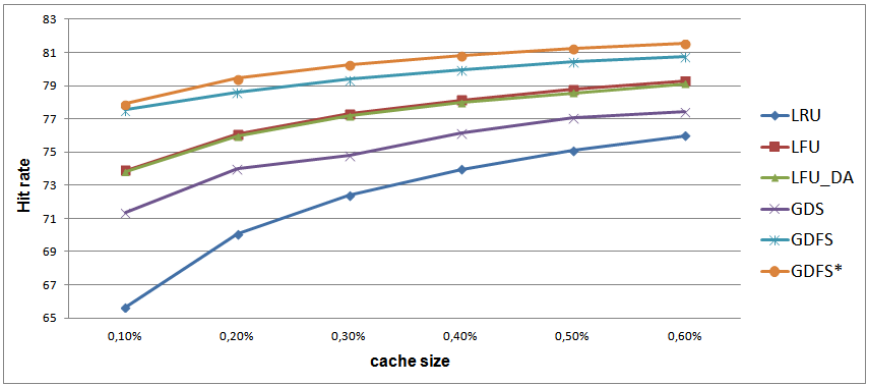
Al revisar en más detalle los resultados, puede observarse que en muchos de los casos el hit ratio es muy alto, donde un con muy poco tamaño se puede alcanzar un hit ratio mayor del 80 %. De hecho, en el caso de rtve.es los hit ratio son más altos que en las otras trazas, dándose el caso de casi un 90 % de hit ratio con tan sólo una caché de 21MB. Esto se explica por el hecho de que esa traza tiene un α **mucho mayor** que las otras, por tanto presenta una concentrada de frecuencia mayor en menos objetos. En las otras trazas, al tener distribuciones similares, otros factores tales cómo el tráfico o el tamaño de los objetos son los que determinan el valor del hit ratio. En cualquier caso, los hit ratios obtenidos son muchos más altos que en otros experimentos, donde el máximo, para un tamaño similar, estaba por debajo del 50 %.

Respecto de las políticas de caché, tanto GDSF como Perfect GDSF han demostrado ser muy eficientes en las pruebas realizadas: como el algoritmo *greedy dual* mantiene los objetos con una alto ratio de frecuencia/tamaño, es capaz de alcanzar un muy alto rendimiento. Hay que tener en cuenta que estas políticas se benefician del patrón de tráfico de las trazas, frecuencia, maximizando la unidad de medida de la caché, el tamaño de objeto. En nuestras pruebas se ha podido comprobar que el número de objetos en la caché en el caso de GDSF era casi el doble del número de objetos que mantenía el LFU, y por eso obtiene mejor rendimiento.

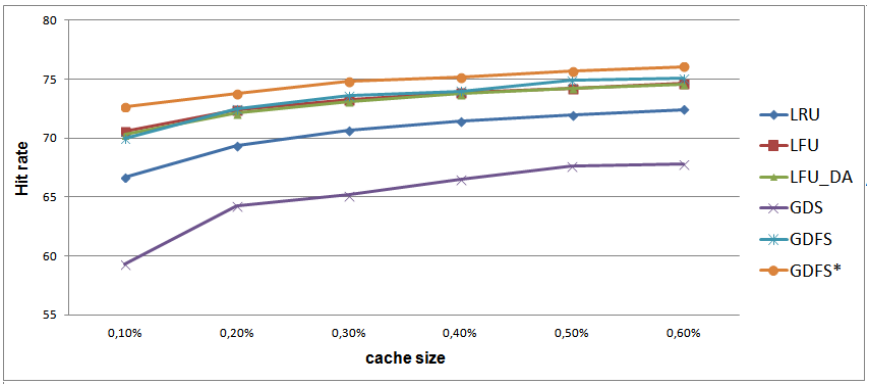
Si se compara el rendimiento entre GDSF y Perfect GDSF, se puede observar que el segundo, al mantener la información histórica de frecuencias, tiene un mejor rendimiento, aunque la diferencia no es demasiado alta. El algoritmo Perfect GDSF es capaz de obtener más de un 70 % de hit ratio con un tamaño de caché de tan sólo el 0.1 % del espacio de trabajo, y llega hasta casi el 90 % con el 0.6 % como tamaño de caché en el caso de las trazas de



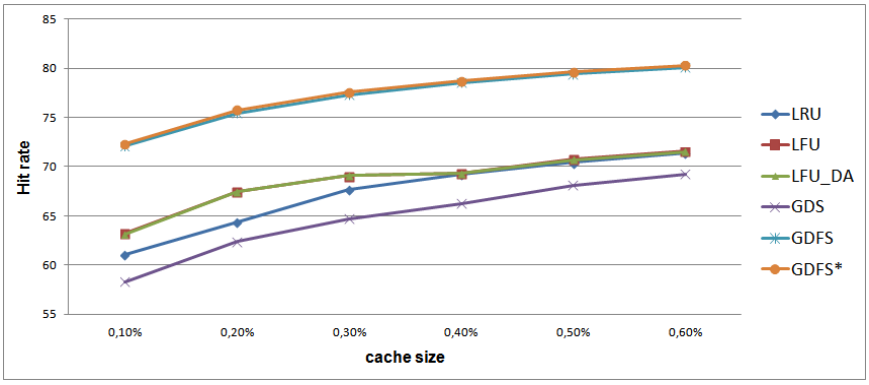
(a) rtve.es



(b) bodasclick.com



(c) eldiario.es



(d) ucm.es

Figura 4.2: Hit ratio obtenido en la simulación de políticas de caché

rtve.es. En las otras trazas se obtiene un hit ratio de alrededor del 80 %.

En el caso del algoritmo GDSF, demuestra ser la segunda mejor política de caché, alcanzando resultados similares al Perfect GDSF. Tiene a su favor que es una política ya existente en productos comerciales y que consume un poco menos de recursos, ya que no mantiene el histórico de frecuencias.

Respecto a los algoritmos LFU y LFU DA, se puede concluir que tienen un comportamiento similar, aunque no es posible determinar cuál ganaría de los dos. Hay que tener en cuenta que las trazas empleadas son de un sólo día y por tanto el efecto del envejecimiento no es tan visible. De cualquier forma, viendo cómo funciona las políticas basadas únicamente en la frecuencia, se puede concluir que al introducir el tamaño como unidad de medida para determinar el número de objetos de caché, estas políticas no lo optimizan y por tanto son superadas por los algoritmos tipo GDSF. Otra conclusión importante es que el rol de la frecuencia es mucho más relevante que el del tamaño. Prueba de ello es que el único basado en tamaño es el que tiene peor comportamiento.

Otra conclusión que se puede obtener es que la políticas basada en el acceso reciente, así como la basada en el espacio tienen un comportamiento volátil. De hecho, en tres de las trazas la política de reemplazo LRU funciona mejor que la GDS, salvo en bodaclick.com donde ocurre lo contrario. Viendo las características de las trazas 4.1, bodasclick.com muestra la mayor concentración de tamaño de objetos grandes entre los más populares, lo cual explica el por qué de su mejor comportamiento comparado con LRU. De cualquier forma, y visto los resultados anteriores, ni LRU ni GDS parecen ser los algoritmos más adecuados para estas trazas.

Si comparamos los resultados con la bibliografía 2.1, encontramos que en líneas generales GDSF mejoraba a la mayoría de las políticas de caché. Esto supone que en principio no hay necesidad de implementar nuevas políticas. De hecho productos como *Squid* (Squid, 2015) lo implementan y no parece tener mucha lógica implementar Perfect GDSF ya que la mejora de rendimiento no es tan sustancial.

Para concluir, tras las pruebas empíricas que se han realizado, se puede establecer que para distribuciones que siguen una Zipf con $\alpha > 1$:

- El rendimiento de la caché es espectacular incluso con un tamaño pequeño. En rtve.es se llega hasta el 90 % con tan sólo 21 MB de caché.
- Para obtener hit ratios elevados no es necesario tener unos almacenamientos que ocupen muchos recursos, ya que con tamaño pequeños se alcanza un hit ratio muy alto.
- Las políticas de caché que mejor funcionan son las que emplean la

frecuencia y maximizan el uso del espacio.

- Los resultados parten de una premisa: los objetos en la caché no caducan. Este aspecto podría ser revisado para completar el estudio.

4.3. Conclusiones

Es este capítulo se ha analizado el impacto, desde el punto de vista de rendimiento de cachés, de que la distribución de peticiones que reciben sitios web sigan una Zipf con muy alto sesgo, esto es, con un valor del exponente alpha mucho mayor que uno.

Este análisis se ha hecho desde dos perspectivas: una formal y otra empírica. Desde el punto de vista formal, una distribución Zipf con un valor de alpha mayor que uno, se convierte en una sucesión convergente, y por tanto acotable. En la sección 4.1 se ha hecho uso de esta característica, lo que ha llevado a completar la fórmula de Shi et al. (2005) para contemplar el caso de $\alpha > 1$. Al dibujar las gráficas del hit ratio en función del valor de alpha se puede ver que a partir del valor 1 (Zipf estricta), el hit ratio crece muy rápido, de tal manera que el hit ratio se dispara. En el lado opuesto, cuanto menor es alpha, menor hit ratio teórico se obtiene.

El resultado anterior se ha validado empíricamente, utilizando logs reales de sitios web, que muestran en valor de α de hasta un 40 % mayor que la Zipf estricta. Como ya se ha comentado en esta tesis, la consecuencia principal es el sesgo tan alto que muestra la popularidad de los objetos web, donde un subconjunto pequeño de objetos reciben la gran mayoría de las peticiones que se reciben.

Esta circunstancia, si es tenida en cuenta, permite que una caché pueda alcanzar un hit ratio enorme sin necesidad de emplear muchos recursos para almacenar en la caché: como ejemplo, en el caso de RTVE, que sigue una Zipf con α de 1,37, almacenando tan sólo 1.000 objetos, lo que supone 20MB de espacio, permite obtener un hit ratio mayor del 84 %. Para un sistema que recibe millones de peticiones al día, 20MB es una cantidad de recursos insignificante, y sin embargo permite incrementar la calidad del servicio de una forma drástica.

Para este tipo de casos, las políticas de cachés a emplear deben utilizar la frecuencia como factor. La política GDSF es una muy buena candidata, ya que ofrece casi los mejores resultados, y está implantada en la mayoría de los sistemas actuales. Sólo ha sido superada por la variación Perfect GDSF propuesta en esta tesis, aunque con resultados casi idénticos.

Finalmente, para las distribuciones con un valor de alpha tan alto, cabe

analizar una mejora adicional en el campo de la coherencia de los contenidos: en el caso de RTVE, aunque de igual forma para los otros, habría que estudiar la mejor estrategia de actualización de esos 1.000 objetos que están en la caché, de tal manera que cada cierto tiempo sean recalculados para evitar que salgan de la caché por estar caducados. Esto permitirá una optimización adicional que, potencialmente, minimizará la carga de trabajo de los sistemas. Hay que tener en cuenta que entre esos 1.000 objetos habrán contenidos estáticos y dinámicos, es decir, algunos que cambien con el tiempo y otros no, y por tanto la estrategia de caché más óptima debería ser sensible al tipo de contenido y a su caducidad.

Tras analizar los efectos en el lado de servidor, en el siguiente capítulo se va a estudiar qué es lo que ocurre en la capa de la vista y si hay alguna correlación entre la vista y el servidor.

Capítulo 5

Zipf en cliente y comportamiento de los usuarios

En los capítulos 3 y 4 se ha abordado la Zipf en el lado del servidor. Las evidencias encontradas muestran que la distribución Zipf es la forma más habitual en el lado del servidor. Además, las evidencias que se han encontrado muestran una mayor concentración de la frecuencia en unos pocos recursos del sitio comparado con los resultados previos de la bibliografía.

En este capítulo se va a mostrar una perspectiva diferente de los servicios web: en vez de analizar el servidor, se analiza el comportamiento en la capa de vista. A partir de los datos de la vista se puede extraer conocimiento sobre el comportamiento de los usuarios.

Cuando se habla de vista nos estamos refiriendo a los navegadores que emplean los usuarios. A este nivel, los elementos que se utilizan para hacer análisis son las páginas web, esto es, las URL de los contenidos de un sitio, así como las interacciones de los usuarios en las páginas. Como ejemplo, en el site de RTVE (rtve.es) las páginas son de noticias que se publican, los vídeos de *alacarta*, encuestas, foto galerías, etc. De hecho, son las páginas web los elementos que se comparten a través de las redes sociales y en definitiva los contenidos principales de un sitio.

El enfoque de vista o servidor hace que se tengan distintas perspectivas de un mismo sitio web. Para ilustrar la diferencia entre ambos enfoques veamos un ejemplo: una página de noticia contiene la información que un redactor a considerado. Además contiene enlaces a otras páginas y demás. Para que la noticia se pueda visualizar necesita imágenes, el logo de la compañía, los CSS, Javascript, etc. En el caso del enfoque de vista, y siguiendo el ejemplo, el único elemento que computa para el análisis en la vista es la noticia, y más concretamente la URL de la misma. Sin embargo, cuando se analiza

en tiempo de servidor, se contabiliza la noticia así como todos los objetos relacionados. Por ello el análisis de vista nos da una perspectiva del usuario y sus elecciones, mientras que la del servidor una perspectiva más técnica.

Para finalizar, debido a las dificultades de obtener información tan sensible para sitios web, se va a hacer un estudio de cómo ha sido el comportamiento de los usuarios de RTVE, un sitio web con mucha demanda de usuarios y que distribuye mucho contenido. Además la perspectiva de la vista intentará ser correlacionada con la del servidor. Esto sólo es posible si para un sitio se tienen los datos de ambas capas, y además los datos incluyen información de varios años.

Este capítulo va a analizar en primer la obtención de datos en la capa de la vista. A continuación se analiza la información y se discuten los resultados del análisis. Finalmente se muestran las conclusiones que se obtienen.

5.1. Obtención de datos en la capa de vista

La obtención de datos sobre cómo se comporta el usuario, qué consume y cómo navega es una tarea que requiere la integración de elementos informáticos dentro de las páginas que los usuarios consumen. Desde el punto de vista del administrador del sitio, esto significa que en todas las páginas se tienen que meter código que permita rastrear lo que el usuario va consumiendo.

Como se comentó en el apartado 2.2.2 la técnica más habitual para hacer estos análisis en la capa de vista es el *page tagging*, que consiste en meter un Javascript en todas las páginas de un sitio de tal manera que tanto en la carga de cualquier página se envía un mensaje a un servicio externo (el servicio de analítica) con la página que se está consumiendo, así como con la identificación del usuario y de la sesión Mahanti et al. (2009). A partir de los datos, el servicio de analítica permite obtener información sobre los contenidos más consumidos, las páginas más vistas, el tiempo promedio de estancia en un sitio antes de que un usuario lo deje o de dónde provienen los usuarios que llegan a un sitio web: de buscadores, de redes sociales, etc.

Como se puede ver esta información es muy sensible, por lo que obtener dicha información es complejo, ya que está dejando al aire los modelos de negocio de las compañías. Dicho lo anterior, sí que hay algunas fuentes de información de dominio público como la que ofrece Oficina de Justificación de la Difusión (OJD) (OJD, 2016), donde se puede consultar el número de usuarios únicos, los tiempos de sesión, etc.

Respecto los servicios de analítica Web, estos proveen dos elementos diferenciados: un sistema de provisión de estadísticas, básicamente un conjunto

Tabla 5.1: Datos analíticas Web

Métrica	(Julio) 2014	(Febrero) 2015	(Febrero) 2016
Usuarios únicos	17.348.642	19.492.087	19.193.320
Visitas	65.453.758	77.741.268	93.937.776
Páginas	327.884.093	364.712.835	363.249.629
Páginas por visita	5,01	4,69	3,87
Duración promedio de visita	11:52	09:52	08:48
Duración promedio en página	02:15	01:52	01:59
% visitas menores de 1 minuto	28,9	28,8	29,4
% visitas entre 1-10 min	30	30,2	30,8
% visitas entre 10-30 min	23,0	23,8	24,3
% visitas entre 31-60 min	17,3	15,9	14,3
% visitas duran más 1 hora	1,8	1,3	1,2
α	1,08	1,12	1,23

de servicios web al que mandar la información descrita arriba, y un sistema de informes con los datos agregados del consumo. Entre otros permite recuperar los datos analíticos que un sitio web lanzó. Los datos de los servicios de analíticas Web son privados y el acceso a los mismos depende de la cesión por parte de los administradores.

En esta contexto hemos tenido la suerte de poder con la cesión de los datos privados por parte de RTVE. Por ello, el análisis de la vista estará centrado en analizar un sitio con un alto volumen de distribución de contenidos, tanto noticias, como vídeos, audios, etc.

RTVE utiliza varios sistemas de analítica web, entre otros Adobe Omniture (Cloud, 2016), quien hace el proceso de recolecta de datos. No es el único sino que además RTVE emplea Comscore y Google Analytics (Google, 2016). Al haber más de un sistema recolectando información es relevante el papel de un tercero que audite los datos, papel que le corresponde a OJD.

El análisis de este capítulo requiere datos de varios años para analizar tendencias en la vista, pero también datos para contrastar hipótesis con respecto el servidor. Por ello se ha seleccionado información en 2014, 2015 y 2016.

Mezclando la información de las fuentes públicas y privadas, y atendiendo a la restricción anterior, se llega a la tabla 5.2 que contiene la información recolectada.

Como se puede ver en la misma, RTVE tiene alrededor de 19 millones de usuario únicos, esto es, usuarios que alcanzan sus servicios al menos una vez al mes. Hay que aclarar que en analítica web el período de análisis suele ser el mes porque permite estabilizar tendencias y minimiza sesgos puntuales.

Con respecto al número de sesiones o visitas se puede ver que ha subido desde los 65 millones a casi 94 millones, lo que supone un incremento de aproximadamente el 50 % de incremento en dos años. Por su parte las páginas vistas presentan una correlación positiva con respecto al número de usuarios, ya que presentan un incremento similar: cuanto más usuarios, más páginas.

Una visita o sesión supone la llegada de un usuario, la navegación dentro del sitio durante un período de tiempo y el consumo de contenidos durante esa visita. Toda visita se caracteriza por dos valores: el número de páginas por visita y el tiempo de visita.

El número de páginas de visita indica el número de distintas páginas que en promedio consume un usuario en una visita. Si atendemos a la tabla 5.2, se puede ver que el número de páginas por visita decrece desde los 5.01 hasta los 3.87. Lo mismo le ocurre al tiempo medio de visita y tiempo media en página. La visita promedio pasó de los 11 minutos y 52 segundos a 8 minutos y 48 segundo en tan sólo dos años. El tiempo promedio en página cayó 15 segundos en dos años, un 11.11 % de descenso. Como se puede observar, hay una tendencia consistente en el sitio web a tener más visitas/sesiones, pero éstas tienen menor duración y en ellas se consumen menos contenido.

El tiempo de visita o tiempo de sesión es un valor que indica cuánto tiempo un usuario dedica a navegar por el contenido de un sitio. En el caso de que nos ocupa y como se ha comentado, este valor tiende a ser menor. Esta tendencia queda remarcada por los porcentajes promedio de consumo en las sesiones. De hecho, las visitas más frecuentes son las que duran menos de 30 minutos, siendo el tipo de visita más usual el que dedica menos de un minuto a consumir el contenido: este comportamiento implica acceder a una página, consumirla y dejar el sitio. A este comportamiento se le llama rebote en la terminología de analítica web. Finalmente, más del 50 % de los usuarios dedican menos de 5 minutos a consumir contenidos en el sitio aunque los servicios de RTVE son primordialmente contenido multimedia.

Finalmente y tras varios intentos, a partir de las evidencias de vista y tras analizar las frecuencias para las página web exportando la información de las herramientas de analítica, un modelo de popularidad para las páginas aparece: la forma de la distribución de frecuencias para las páginas web de RTVE sigue una Zipf 2.3, esto es, la frecuencia de una página web depende del ranking de popularidad. Utilizando de nuevo el estimador de Clauset et al. (2009) y la librería powerlaw de R (Gillespie, 2014), se ha determinado el valor de α para las distribuciones de 2014, 2015 y 2016. Estos resultados aparecen en la última fila de la tabla 5.2, y la forma característica de tipo Zipf se puede ver para cada año en la figura 5.1.

Analizando el valor de α se puede ver que el valor de α se está incre-

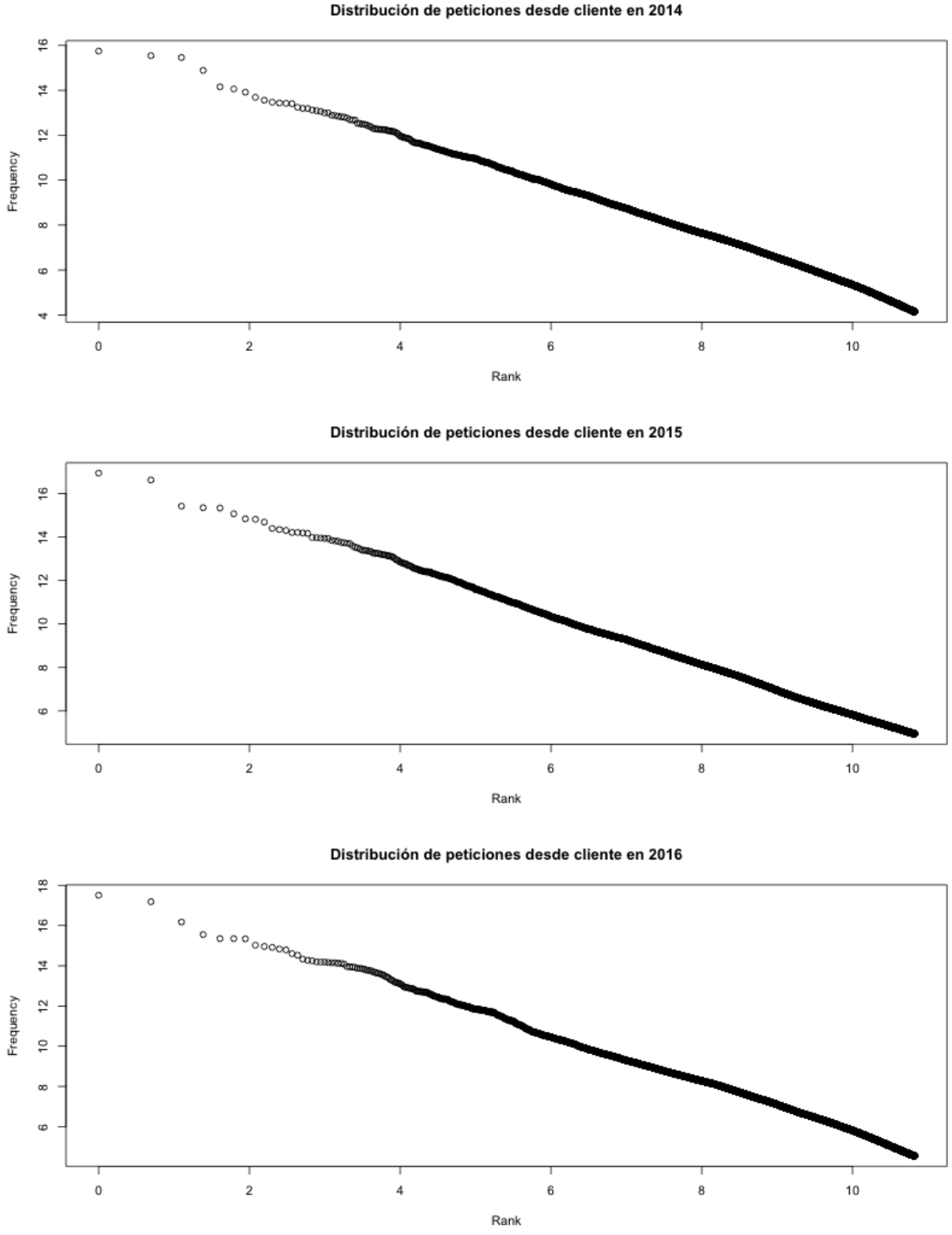


Figura 5.1: Figura Zipf en los logs de cliente en 2014, 2015 y 2016

mentando con el paso de los años, pasando de 1,08 hasta 1,25, o lo que es lo mismo, un incremento del 15,75 % en sólo dos años, y esto a pesar de tener más usuarios. Este incremento implica una mayor concentración de frecuencias en cada vez menos páginas web. Desde un punto de audiencias, y teniendo en cuenta que estamos en la capa de la vista, se puede afirmar los usuarios cada vez consumen más, menos contenidos y que la audiencia se concentra en una serie de ellos. El resto de contenidos pasan a ser marginales con frecuencias pequeñas.

Analizando otros sitios utilizando OJD (2016), estas tendencias también aparecen en otros sitios web. Como ejemplo, telecinco.es, otra web de una televisión, esta vez privada en España, presenta un patrón similar si comparamos 2014 con 2016: pasó de 62.275.280 a casi 76 millones de sesiones, el tiempo de sesión pasó de 6 minutos y 24 segundos a 4 minutos y 32 segundos y el número de páginas por sesión de 3,43 a 3,1. Como se ve, las tendencias que se pueden ver en los datos públicos muestran un patrón similar en los sitios web auditados por OJD (2016) (canalsur.es, ojd.es, telemadrid.es, etc.).

Para ver el efecto que ha podido generar en la capa de servidor las evidencias de la capa de vista, es necesario tener datos de los mismos períodos en la forma de access.log (ver capítulo 3). En la siguiente sección se muestran las características de los logs recopilados para los mismos períodos en el sitio rtve.es.

5.2. Logs de servidor para las evidencias de vista

Tal y como se ha comentado en el apartado anterior, un conjunto de logs con el formato CLF en la forma de access.log se han recogido para las fechas que aparecen en la tabla . Para procesar los ficheros y poder hacer los cálculos para caracterizar las distribuciones ha habido que procesar un gran volumen de ficheros. De hecho, los logs de RTVE involucran hasta un total de 6.000 millones de URLs.

Los logs se han procesado siguiendo el proceso indicado en la sección 2.2.1 y se ha calculado las frecuencias para cada uno de los objetos web. A continuación se ha utilizado el estimador de Clauset et al. (2009) y la librería powerlaw de R (Gillespie, 2014). En la tabla 5.2 se presentan los resultados incluyendo α y el número de peticiones promedio por día.

Como se puede ver, el servicio de rtve.es duplicó el número de peticiones diarias a servidor de 2014 a 2015 y subió un 14,6 % de 2015 a 2016.

Respecto al valor de α, siempre es mayor que 1. Además éste se incrementó

Tabla 5.2: Logs de servidor para evidencias de vista

Período	Peticiones diarias por día	α
Julio 2014	130 millones	1,41
Febrero 2015	288 millones	1,43
Febrero 2016	330 millones	1,33

un poco de 2014 a 2015, pero en 2016 se notó una bajada. En 2016 RTVE hizo un cambio en sus páginas para que se adapten al dispositivo (*responsive design*), con lo que la página cambia según el dispostivo. Este cambio no afecta a las URL de la vista, sino a cómo se pintan las mismas. La técnica que se usa para adaptar los contenidos se basa en el uso de redirecciones HTTP en función el agente de usuario (Fielding et al., 1999). Esto hace que para un objeto web utilizado en una página web, la URL que la resuelve depende del dispositivo, creando más URLs. El efecto es una disminución de α ya que la frecuencia de los objetos que aparecen en las páginas web se dividen en las versiones para cada tipo de dispositivo.

A continuación, tras ser presentados los datos se analizan los resultados y se correlacionan la vista y el servidor.

5.3. Análisis de las evidencias de la vista

Para analizar las evidencias de la vista es importante contextualizar el sitio web. RTVE ofrece contenido de audio y vídeo que proviene de la emisión de televisión y de la radio. La duración promedio para una emisión de radio es de una hora y de televisión es alrededor de 50 minutos. Sin embargo, el contenido de televisión de prime time (el contenido de televisión más importante) tienen una duración promedio de 90 minutos. Es importante notar que este tipo de contenido es el más demandado en la tele, pero si analizamos los datos de duración promedio de la tabla 5.2, los contenidos de televisión no son ampliamente consumidos en la web por los usuarios. De hecho, tan sólo el 1,2 % de las visitas duran más de una hora, con lo que el máximo número de usuarios que ven programas del prime time completos es de 1.116.000. Esto es importante para hacer notar que el consumo de la televisión y el consumo de la web no son equivalentes.

A lo anterior habría que añadir que en la web no sólo se ofrece el contenido de la televisión, sino contenido adicional diferente. Además de ello la web ofrece un acceso libre a los contenidos. Es un hecho que al menos el 50 % de los contenidos que se consumen en los sitios viene de buscadores como

Google y las páginas se plantean para facilitar las búsquedas de los usuarios. En la web el usuario goza de una libertad para consumir que no existe en la televisión y de ahí esta diferencia.

Como se comentó en la sección 5.1, el tiempo de sesión, para el sitio rtve.es, baja desde los 11 minutos y 52 segundos hasta los 8 minutos y 48 segundos, un 25,84 % de descenso. Además de lo anterior, hay que tener en cuenta que casi el 50 % de las visitas son menores de 5 minutos, y casi el 30 % duran menos de un minuto. Si analizamos estos datos desde la perspectiva del usuario, implica que cuando un usuario llega a un sitio web, tiene un 30 % de probabilidades de irse a otro sitio a consumir y no continuar en rtve.es. Este comportamiento, si se compara con los años anteriores, está creciendo y las sesiones están siendo cada vez menores.

El comportamiento antes mencionado se ve reforzado por el hecho de que el número de páginas por vistas ha decrecido pasando de 5,01 a 3,87: el usuario que accede a un sitio web entra de forma más directa al contenido que quiere consumir, e invierte menos tiempo en moverse por otros contenidos del sitio. Desde un punto de vista sociológico, en RTVE se piensa que esto es consecuencia de tres factores: la movilidad, las redes sociales y finalmente los motores de búsqueda.

El usuario en móviles tiene un comportamiento más compulsivo y gasta menos tiempo en consumir contenidos adicionales. Las redes sociales se emplean como altavoces de los contenidos que se producen. De hecho el consumo en redes como twitter es muy impulsivo y el contenido de rtve.es accedido desde redes sociales sigue una Zipf. Finalmente, más del 50 % del acceso de los usuarios vienen desde motores de búsqueda y estas sesiones presentan una alta tasa de rebote: el usuario que busca un contenido, lo consume sin más en más de un 40 % de los casos.

Dos datos refuerzan la conclusión anterior: el primero es el hecho del descenso en el tiempo en página, que en tan sólo 2 años, del 2014 al 2016, ha bajado en un 50 %. El otro proviene de OJD (2016). En el sitio se puede ver que el usuario en móvil invierte sólo 3 minutos y 40 segundos por visita, menos de la mitad del promedio de RTVE, lo que implica un acceso más directo desde móvil.

Respecto la correspondencia entre páginas vistas y páginas web visitadas, en la tabla 5.2 se puede ver que la distribución de frecuencia de las páginas web que los usuarios visitan siguen una distribución tipo Zipf. La distribución, como se vio en la sección 2.3, está dirigida por α. Cuanto mayor es su valor, más escorada está la distribución y mayor concentración de frecuencias hay en menos objetos. Se puede observar que la popularidad de los objetos que se consumen en la vista es cada vez más importante, y de hecho, el valor

ha crecido un 13.88 %. El factor que explica este aumento es la elección del usuario, máxime cuando se ha producido un aumento de la Zipf en 2016 frente a 2015, cuando el número de usuarios fue similar. Esto implica que ante el mismo número de usuarios, al año siguiente consumieron de forma más asimétrica.

Desde el punto de vista técnico, esto es, viendo la parte del servidor, el patrón de consumo de las páginas web debería modificar la distribución de frecuencias de los objetos web. Se puede observar que el incremento de α en el cliente en 2015 provocó un incremento del α en el servidor. Sin embargo, el nivel de incremento en la vista es mayor que en el servidor, de hecho el incremento en el cliente ha sido del doble que en el servidor. Esto se justifica por el efecto *trickle down* Doyle et al. (2002): según dicho principio, un nivel de caché implica un menor α en la siguiente capa. Hay un factor clave que tiene que ser considerado: la cache de cliente. Un navegador cachea peticiones para mejorar el tiempo de carga, y dicho cacheo provoca un suavizado de las frecuencias que llegan al servidor.

También desde el punto de vista técnico, los datos muestran un descenso de α en el servidor entre 2015 y 2016 de un 7 %, a pesar de que crecieron en la vista. Nuestra hipótesis para explicar esto es el efecto del rediseño de RTVE y la distribución de las frecuencias de los objetos en servidor entre las distintas URLs dependiendo el dispositivo. Esta hipótesis está soportada por el gran incremento de peticiones diarias en promedio que se produjo, pasando de los 288 millones en 2015 a los 330 millones sin incremento en el número de usuarios.

5.4. Conclusiones

El análisis comparativo de la vista con los datos obtenidos de RTVE y contrastado con las consecuencias en el servidor ha sido el propósito de este capítulo. Para ello se han comparado evidencias reales de los años 2014, 2015 y 2016 en un sitio tan demandado como el de rtve.es.

El objetivo ha sido analizar el comportamiento del usuario en la capa de la vista, cómo ha evolucionado y evaluar si ha habido impacto en el servidor.

Las principales conclusiones a las que se ha llegado son:

- Como ya se ha apuntado en el lado del servidor, en el lado de cliente, las evidencias demuestran también una gran relevancia de la popularidad de los páginas web. De hecho, la distribución de frecuencias de las páginas web sigue una distribución Zipf, donde α está incrementándose.
- A partir de los datos del cliente, la tendencia de consumo del usuario

apunta a un modo de consumo más directo al contenido, y una menor influencia de los webmasters sobre los usuarios. Esto está soportado por varios hechos: hay más visitas, éstas duran menos y consumen menos recursos por visita y finalmente la distribución de la popularidad es cada vez más dispar.

- El consumo desde el móvil, ya que el tiempo de consumo en móvil es mucho menor que en otros dispositivos, el acceso desde las redes sociales y el uso desde los buscadores son lo que motivos que explican el modo de consumo compulsivo del usuario.

- Hay una correspondencia, si no cambian los conceptos técnicos de un sitio, entre el incremento de α en cliente y en servidor. El incremento en el cliente parece ser mayor que en el servidor.

El hecho de encontrar correlación entre cliente y servidor lleva a la cuestión de si dicha correlación siempre existe, y de si la Zipf en servidor es una consecuencia de que se dé en el cliente, como se ha evidenciado en las evidencias. El capítulo siguiente tratará de responder a la cuestión aplicando un enfoque diferente: utilizar la simulación de la distribución en el cliente y validar qué distribución se consigue en el servidor.

Capítulo 6

Simulación de distribución de clientes y efecto en servidor

En el capítulo 2 se comentó que el enfoque de esta tesis era sobre todo analítico, donde tras la adquisición de los datos se analizan para obtener una conclusión más o menos relevante. Pero también se comentó sobre lo eficaz que puede ser la simulación para testar episodios o situaciones que no existen o no se pueden tener.

En el capítulo anterior se han analizado evidencias de la presencia de distribuciones de frecuencia que siguen una Zipf. Dichas distribuciones aplicadas al dominio de la web, y a la capa de la vista, nos han permitido entender y modelar el comportamiento de los usuarios: estos interaccionan con un sitio web navegando por las páginas durante un tiempo y consumiendo un conjunto finito de objetos. Así, en la capa de la vista el comportamiento de un usuario viene caracterizado por cómo son las sesiones de navegación, y éstas por las páginas que se consumen en ellas y el tiempo de consumo.

Respecto a esas variables, y tras su análisis, se ha podido comprobar que la tendencia, en sitios como rtve.es o telecinco.es, es de decrecimiento. Este puede ser provocado por diversos factores: redes sociales, movilidad, etc.

El comportamiento en la capa de la vista tiene una influencia directa sobre lo que ocurre en la capa de servidor: así hemos visto que, sin cambio de tecnología o procesos, un sitio web que tiene un cambio de comportamiento en cliente muestra ese cambio en el servidor. El ejemplo de RTVE es claro, ya que subió α en cliente y se vio dicho cambio en el servidor.

Esto nos lleva a plantear una hipótesis de trabajo: la distribución de frecuencias de las peticiones que recibe un sitio web siguen una distribución Zipf porque las páginas que consumen los usuarios en la capa de cliente también siguen dicho tipo de distribución. Si esta hipótesis fuese cierta, significaría

que el usuario es quien provoca la Zipf en el servidor a través de sus acciones en el cliente. Por ello, cabe preguntarse si en el servidor ocurriría una Zipf si en el cliente no se da dicha distribución.

Este capítulo intentará refutar o validar esta hipótesis. Para ello se va a realizar una simulación contra dominios web para validar si para distintas distribuciones, una sitio web genera o no una Zipf en el servidor. Por ello, en este capítulo primero se introduce el problema, luego se planteará el modelo de simulación, a continuación se planteará la propia simulación y se mostrarán los resultados. Al final se presentarán unos resultados que permiten responder a la cuestión planteada.

6.1. Definición del problema

Como se ha indicado en la introducción, en esta sección se va definir la hipótesis que se pretende aclarar mediante simulación. La definición del problema es relevante porque va a permitir entender cómo será resuelto.

A partir de las trazas y evidencias, que se han recogido en los distintos capítulos, se llega a que la distribución de la frecuencia sigue una Zipf tanto en la capa de la vista como en la del servidor. Hay muchísimas evidencias en el lado de servidor (Krashakov et al., 2006; Zotano et al., 2015b), con lo que parece justificado plantear si el origen de dicho comportamiento está en las acciones que ocurren en la capa de la vista. Hay que recordar, tal y como se comentó en el capítulo anterior, que las acciones de los usuarios en la capa de vista tienen un refrendo en el lado de servidor, ya que las primeras son la fuente de las peticiones que se registran en servidor.

Hay dos formas de intentar explicar si las acciones de los usuarios son las que provocan la existencia de las Zipf en servidor: o por negación, es decir, encontrar un caso de distribución en la vista que resulte en una distribución distinta de la Zipf en el servidor. De existir, significaría que para sitios donde pueden darse distribuciones no Zipf, esto no ocurre, y por tanto la justificación sería el comportamiento del usuario. La otra alternativa es asumir que un sitio siempre genera una Zipf para el servidor, con independencia de cuál es la distribución en cliente.

El primer enfoque es difícil de implementar, ya que toda simulación va a requerir una masa crítica de páginas web y generar tantos juegos de pruebas como sea necesario hasta encontrar uno que permita afirmar que hay distribuciones que permiten un modelo de servidor que no sigue una Zipf. De hecho el horizonte potencial de juegos de prueba es infinito: habrá que generar distribuciones más o menos largas hasta conseguir una que no de una

Zipf.

Sin embargo el segundo modelo es mucho más sencillo de implementar. La idea de la que se va a partir es que un sitio web es, a efectos de distribución de frecuencias, un transformador desde una distribución de páginas web en cliente a una función de distribución en el servidor. Si todas las distribuciones de cliente que se prueben dan una Zipf, no significa que sea verdad que un sitio web sea un transformador, pero nos dará un resultado relevante. De cualquier forma, los datos que se puedan obtener con este enfoque complementarán el conocimiento sobre el por qué ocurre una Zipf en el servidor. Hay que tener en cuenta que en la bibliografía, en las referencias de Zipf en servidor, no hay información sobre cómo era la distribución en cliente que la ocasionó, por lo que hay una carencia de estudios en este sentido.

Planteando formalmente la hipótesis, sea PW el dominio de las páginas web un sitio web, sea OW el dominio de los objetos web del mismo sitio, entonces asociada al sitio hay una función $site$, tal que:

$$site(dv) = ds \tag{6.1}$$

donde dv es un conjunto de elementos de PW, y ds es un conjunto de elementos de OW que resultan de las peticiones de los objetos de dv y donde la distribución de las frecuencias de ds sigue una Zipf con un valor de α.

Dado el problema definido en este punto, en el siguiente se analizará cómo será el modelo de simulación.

6.2. Modelo de simulación

En el apartado anterior se ha definido cuál es la hipótesis de trabajo: un sitio web es una función que dada una distribución en la capa de usuario, devuelve una Zipf en el servidor (ver ecuación 6.1).

Para poder montar una simulación que permita obtener valores significativos, es necesario determinar los siguientes elementos:

- Sitios web que permitan hacer la simulación.
- De un sitio web, qué URLs se van a utilizar para hacer el test. Lo lógico es que a partir de la home se hiciese algún tipo de recolección de páginas para tener el conjunto PW.
- Determinar qué distribuciones en la capa de la vista se van a emplear a partir de las páginas web seleccionadas.

- Determinar el modelo de navegación. La prueba no puede ser únicamente un listado de URLs, sino que debe permitir la inclusión del concepto de sesión. Hay que recordar (ver capítulo 5) que una sesión está definida por el conjunto de páginas que se pide en ella.
- Recolecta de los datos que llegan a servidor.

Para la simulación lo primero es determinar los dominios que se van a emplear. Uno de los sitios que va a ser utilizado es rtve.es, que ha sido objeto del análisis en el capítulo 5, por lo que se puede corroborar los datos que se obtienen con las evidencias anteriores. Para tener otra visión de un sitio distinto, no necesariamente basado en multimedia y de propósito general, se ha seleccionado la wikipedia (http://wikipedia.org). Estos dos sitios presentan características similares: son altamente demandados, siguen una Zipf en el servidor (Urdaneta et al., 2009; Zotano et al., 2015b) e incluyen contenido de todo tipo: vídeo, fotos, etc.

Una vez determinado el sitio web, el siguiente apartado es la recolección de páginas web. Para la simulación de ambos dominios se va a tomar un número de páginas web, que en el caso de esta simulación se establecerá en 1.000. Estas páginas se obtendrán de forma aleatoria haciendo una navegación por el sitio, seleccionando las URLs hasta completar las 1.000.

Con las páginas web determinadas y para poder crear las distribuciones de peticiones, es necesario acotar el número de las mismas. Por ello se ha seleccionado que el número de peticiones a esas 1.000 páginas será 10.000, 20.000, 30.000 y 40.000. Así, las distribuciones estarán acotadas a ese número de peticiones, con lo que es más sencillo crearlas. Esta selección va a permitir tener resultados diferentes y evaluar cómo evoluciona la distribución del servidor a medida que sube el número de peticiones.

Respecto de las distribuciones que se van a utilizar para validar el comportamiento en la capa de vista, la selección incluye Zipfs, y otras distribuciones estándares. A continuación se listan las distribuciones que se van a emplear:

- Zipf con α igual a 0,7. La distribución de frecuencia de las páginas web siguen una distribución Zipf con un valor de α igual a 0,7.
- Zipf con α igual a 1. La distribución de frecuencia de las páginas web siguen una distribución Zipf estricta.
- Zipf con α igual a 1,3. La distribución de frecuencia de las páginas web siguen una distribución Zipf con un valor de α igual a 1,3.
- Distribución aleatoria. Cada página web se pedirá un número aleatorio de veces.

- Distribución normal. La distribución de frecuencia de cada página web sigue una distribución normal.

- Distribución uniforme, es decir, todas las páginas web reciben el mismo número de peticiones.

El modelo de navegación implica el modelo de sesión que se empleará. En el caso de esta simulación sólo se va a contemplar el número de páginas por sesión, es decir, el número promedio de páginas por visita. El hecho de que cada visita tenga un número de páginas distinto hace que la simulación sea más realista. En un sitio web las sesiones son diferentes y por tanto en la simulación se va a hacer algo similar. El valor promedio del número de páginas por visita seleccionado es de 3,5, valor muy similar al de rtve.es (ver capítulo 5).

Finalmente, el asunto más complejo es el de obtener el log de servidor, esto es, el conjunto de peticiones que el servidor recibe. Hacer estas pruebas en conjunto con los administradores de rtve.es y wikipedia.org es algo demasiado complejo. Sin embargo es sencillo determinar el log de servidor, si los navegadores están bajo control. Al final, todo el log de servidor proviene de las llamadas de los clientes, por lo que bastará con registrar todas las peticiones que se hacen en los navegadores bajo simulación.

En el punto siguiente se muestran los valores de las simulaciones y se comentan los resultados obtenidos.

6.3. Simulación y resultados obtenidos

Siguiendo el modelo de simulación comentado en la sección anterior, se ha desarrollado un sistema basado en agentes para realizarla. Dicho sistema está parametrizado por cuatro valores: número de páginas web a testar (npw), dominio, número de peticiones ($npet$) y función de distribución que seguirán las peticiones ($fdis$).

Con los parámetros anteriores, lo primero que hace el sistema es consultar un dominio, navegar por sus páginas y seleccionar npw páginas web del mismo. A partir de estas páginas, el sistema genera la distribución de peticiones siguiendo la distribución $fdis$ acotada al número de peticiones $npet$. Para ello, utiliza un sistema estadístico, R, determinando en primer lugar el número de repeticiones que tiene que darse para npw elementos y creando un vector de frecuencias que cumple con la distribución. Luego, aleatoriamente, asigna un número del vector a cada página web, con lo que se crea un vector de páginas web.

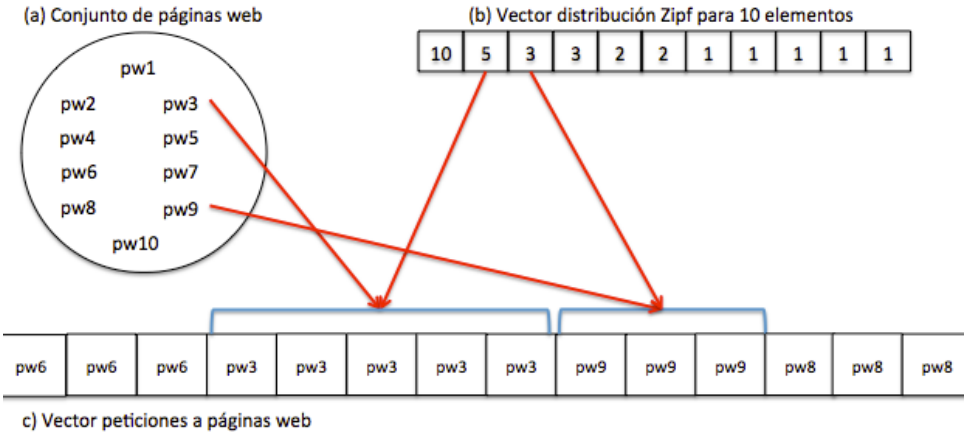


Figura 6.1: Ejemplo de funcionamiento a la hora de generar el vector de peticiones

Este proceso se ilustra en la figura 6.1. En ella *npet* es igual a 30, *npw* es igual a 10 y $fdis$ es una Zipf con α igual a 1. Como se puede ver en ella, se ha hecho una selección de 10 páginas web (a), y se ha generado un vector de frecuencias (b). A continuación para cada objeto de (a) se le asocia un número de (b), que será su número de peticiones. Esta relación es biunívoca para garantizar que las peticiones que se harán sean exactamente *npet*. Con estos dos elementos se crea el vector de peticiones (c) replicando tantas veces un objeto como peticiones tenga que realizar. En el ejemplo $pw3$ aparece 5 veces en el vector de peticiones ya que 5 ha sido la frecuencia que se le ha asociado.

Con el conjunto de peticiones se va a construir las sesiones que se emplearán. Tal y como se ha establecido en la sección anterior, el número de páginas promedio de una sesión será de 3,5. Para ello, a medida que se construye una sesión, ésta se completa con páginas web elegidas aleatoriamente del vector de páginas. Toda página que es incluida en una sesión es eliminada del vector de peticiones hasta terminar la prueba, que será cuando ya no queden más páginas en el vector.

La implementación del sistema de simulación se basa en el paradigma de agentes utilizando *Mason* (Luke et al., 2005) como framework de desarrollo. La idea es que un agente simule el comportamiento de un usuario. Por ello, un agente toma como parámetro una sesión que tiene que navegar. Las sesiones se generan como se comentó anteriormente y son creadas por otro agente que tiene como rol generarlas.

El agente que simula un usuario necesita realizar las llamadas web de la

Tabla 6.1: Valor de α para la simulación en rtve.es

Páginas	Objetos	Zipf 0.7	Zipf 1	Zipf 1.3	Aleatoria	Normal	Uniforme
10.000	1.245.421	1,16	1,52	1,82	0,98	0,87	0,65
20.000	2.783.609	1,39	1,56	1,72	1,16	0,98	0,64
30.000	3.915.142	1,15	1,33	1,65	1,18	0,89	0,64
40.000	4.845.070	1,18	1,33	1,63	1,14	0,97	0,71

sesión. Para ello se emplea un navegador sin interfaz de usuario, esto es, un navegador que realiza todas las acciones pero sin presentar una pantalla de visualización. Para este caso, el navegador que se ha usado para implementar la solución está basado en *HtmlUnit* (Bowler, 2002), aunque ha sido modificado para garantizar que todas las peticiones que realiza sean enviadas a un sistema de almacenamiento. Dicho almacenamiento permite disponer de los datos para que un tercer tipo de agente sea capaz de analizar cuál es la distribución que se va generando, y si es el caso, si corresponde a una Zipf. Para realizar dicha validación se emplean los mismos mecanismos que se describieron en la sección 2.3, esto es, el estimador de Clauset et al. (2013) y la librería powerlaw de R (Gillespie, 2014).

Con la implementación descrita se han llevado a cabo las simulaciones comentadas en la sección 6.2. En el caso de rtve.es, los resultados que se han obtenido se pueden observar en la tabla 6.1.

Observando los datos de rtve.es se puede ver que en todas las simulaciones obtenemos una Zipf, con independencia de cuál sea la distribución en el cliente. Se puede ver que esto ocurre incluso con valores muy pequeños de peticiones. Hay que tener en cuenta que el número de objetos únicos que distribuye rtve.es es alrededor de los cuatro millones. Por ello, al menos en rtve.es, podemos concluir que incluso un volumen relativamente pequeño de peticiones es capaz de generar una Zipf estable en servidor. Hay que tener en cuenta que el volumen de peticiones en servidor ha sido del 1,5 % del tráfico real que llega a los servidores de rtve.es (en el caso de 40.000 peticiones de cliente).

Además de lo anterior se observa cómo el valor de Zipf se estabiliza a medida que sube el número de peticiones en todas las distribuciones. Este hecho ya fue descrito por Roadknight et al. (2000), quién descubrió que a medida que sube el número de peticiones el valor de α se estabiliza. Este hecho es relevante, ya que puede permitir valorar con subconjuntos de peticiones cuál es la tendencia de un sitio, o si un sitio está cambiando su distribución. En cualquier caso, el valor de α obtenido con este tamaño de muestra puede presentar mucho sesgo por cómo se ha seleccionado el conjunto de datos. Con

Tabla 6.2: Valor de α para la simulación en wikipedia.org

Páginas	Objetos	Zipf 0.7	Zipf 1	Zipf 1.3	Aleatoria	Normal	Uniforme
10.000	461.965	1.09	1.28	1.39	0.74	0.62	1,14
20.000	910.035	1,11	1.43	1.54	0.67	0.67	1,41
30.000	1.258.111	1,09	1.35	1.49	0.61	0.65	1,43
40.000	1.745.229	0,89	1.22	1.43	0.62	0.63	1,4

independencia de lo anterior, el valor de α encontrado para las Zipfs no está muy lejos de los valores reales que se mostraron en el capítulo anterior. Como ejemplo, en 2015 se encontró un α de servidor de 1.43 cuando en el cliente la distribución tenía una α de 1.12. En este caso, con una Zipf estricta el valor en servidor es de 1.33.

Al hacer una comparativa entre las distintas Zipfs en cliente, se puede observar que cuanto mayor es el valor de α en cliente, mayor es el valor que se observa en el servidor. Esto va en línea con los datos que se mostraron en el capítulo anterior, donde un incremento de la concentración de popularidad en el cliente provocó un incremento en el servidor.

Respecto a las otras distribuciones, resalta la relativa estabilidad de la distribución aleatoria, aun cuando las sesiones no son equivalente entre cada una de las pruebas que se han hecho. En el caso de la normal y la uniforme también se observa la estabilidad en los datos obtenidos. Es sorprendente que esto ocurra sobre todo en la distribución uniforme: cómo se generan los one-timers (peticiones que sólo se hacen una vez al servidor). Analizando el log se puede ver que hay muchos muchas urls que se basan en un timestamp, o en peticiones personalizadas para la sesión. De ahí es donde surge la cola de la distribución.

Por su parte, en la tabla 6.2 se puede observar los resultados de la simulación para la wikipedia.

Se puede observar en estos datos obtenidos simulando wikipedia.org que, al igual que ocurre en rtve.es, se obtiene una Zipf, con independencia de cuál sea la distribución en el cliente. El valor de la Zipf se estabiliza bastante pronto y no hace falta un gran volumen de peticiones de cliente para obtener esta estabilidad. Además, al igual que antes, al aumentar el valor de α en cliente, mayor es el valor que se observa en el servidor.

Cabe resaltar que en el caso de la wikipedia, el valor de α en servidor, cuando en cliente se da una Zipf, es menor que en el caso de rtve.es. Sin embargo, esto no ocurre en el caso de distribuciones estándar. Destaca también que en el caso de un acceso uniforme, la wikipedia presenta una Zipf que dobla a rtve.es.

También, si comparamos ambas simulaciones se puede observar que una petición en rtve.es genera algo más de 120 objetos web, mientras que en el caso de la wikipedia, el número de objetos web generados es de algo más de 40 objetos web por petición de cliente. Al intentar correlacionar el tamaño de objetos de servidor con el valor de α en servidor, no se ha encontrado un resultado positivo. La conclusión es que el número de objetos no influye sobre la distribución de probabilidad de la frecuencia.

Si analizamos las dos simulaciones, se puede ver que en todos los casos hemos encontrado una Zipf, a pesar de que hemos variado la distribución de los objetos en la capa de cliente. El hecho de que, con independencia de las acciones de los usuarios, en el servidor siempre aparezca una Zipf, permite descartar que sean las acciones del usuario las que provocan la existencia de una Zipf en servidor. Esto nos lleva a que, teniendo en cuenta la cantidad de evidencias a favor de la existencia de Zipf en servidor, y nuestras simulaciones, el hecho de que una Zipf ocurra en servidor debe estar relacionado con algo estructural de las páginas web, más concretamente, a cómo se articulan los recursos dentro del HTML. La arquitectura de página, el modelo de llamada a recursos dentro de las páginas y el modelo de reutilización de objetos web dentro de un sitio tienen que ser los elementos que hacen que las peticiones de servidor acaben teniendo una distribución tipo Zipf. Los usuarios, en lo único que afectan es en cómo será la pendiente de la distribución, es decir, en que el valor de α sea mayor o menor.

Finalmente, las simulaciones que se han desarrollado sugieren que, desde el punto de vista de las distribuciones en cliente y servidor, un sito web es únicamente un transformador de una distribución en cliente a una Zipf en servidor, tal y como se expresaba en la hipótesis de trabajo y en la fórmula 6.1.

6.4. Conclusiones

En este capítulo se ha intentado analizar si el hecho de que ocurran Zipfs en servidor es una consecuencia de los usuarios, o si es una característica de las páginas web. Para ello se ha definido el problema que se plantea, así como un modelo de simulación para validar la hipótesis de que un sitio web es únicamente un transformador de distribuciones de usuarios en Zipfs. A continuación se ha implementado un sistema basado en agentes para hacer la simulación y se han obtenido unos resultados. Tras su análisis destacan las siguiente conclusiones:

- Para rtve.es y para la wikipedia, con las distribuciones planteadas siem-

pre se da una Zipf. Además los valores que se han encontrado son estables y siguen las pautas de las evidencias encontradas.

- El comportamiento del usuario en la vista no es el causante de que en servidor se de una Zipf. De hecho, distribuciones uniformes también generan Zipfs en servidor. El papel del usuario es determinar con su comportamiento cuál es el α de la distribución.
- Descartado el usuario, la distribución de Zipf en servidor debe estar motivada por alguna, o todas las siguientes razones:
  - La arquitectura de las páginas, esto es, qué elementos aparecen en las páginas.
  - Modelo de reutilización de recursos entre páginas.
  - El modelo de generación de páginas que marca HTML y su forma de incluir dependencias entre recursos.
  - El modelo de llamada a recursos dentros de las páginas. Este último sería el que provocaría que haya tanto one-timer.

Capítulo 7

Conclusiones y trabajo futuro

Esta tesis arrancó con la hipótesis de que los sistemas de caché serían más eficientes en consumo de recursos si se tuviera un mejor conocimiento de lo que solicitan los usuarios. Sobre dicho conocimiento, también se asumía que la frecuencia de aparición de solicitudes de objetos web seguía una función Zipf. Bajo estos dos supuestos, la tesis ha confirmado las hipótesis de partida y avanzado en el conocimiento de la forma de la función Zipf en sistemas modernos.

La contribución principal de esta tesis son las evidencias que siguen avalando la función Zipf como modelo que caracteriza la función de distribución de la frecuencia de las peticiones de usuarios en dos ámbitos. Primero, concibiendo las peticiones a bajo nivel como peticiones a los objetos web en el lado del servidor. Segundo, y como novedad, con un análisis similar realizado usando páginas web solicitadas desde el lado del cliente. La confirmación de la existencia de Zipf se hace tras la compilación y análisis de un banco de datos superior al de trabajos identificados en la literatura en su variedad de sectores considerados, tecnologías evaluadas, rigor en la obtención de los datos y en volumen. Ello da mayor respaldo a la hipótesis de la función Zipf comparado al que había hasta ahora. Por ello, se puede concluir que la función Zipf ocurre en los sitios web estudiados con independencia del sector del sitio web, su tecnología o propósito.

Además, tanto en el lado del cliente como en el del servidor, se ha encontrado que el valor α que caracteriza dicha función Zipf es mucho mayor que en el encontrado en la literatura, hasta un 87 % mayor en el lado del servidor y un 43 % en el lado del cliente. Esta información tiene importantes consecuencias para aquellos sistemas de información masivos que deben atender a millones de usuarios.

Finalmente, con el conocimiento adquirido, se ha intentado abordar la

cuestión del origen de las Zipfs, concluyendo que las evidencias señalan que podría ser algo inherente a la estructura de las páginas web y no asociado al comportamiento en sí del usuario. También que hay correlación lineal entre el valor α observado en el lado del cliente y en el del servidor. La magnitud de α podría cambiar en el futuro cercano, pues se ha encontrado que nuevas técnicas de diseño web conllevan menores valores de α.

Los resultados de la tesis han sido presentados en varios congresos y dos revistas que pueden consultarse con mayor detalle en el apéndice A. Destacan:

- La construcción de simulaciones para evaluar el impacto en cachés se publicó en el las IX Jornadas de Ciencia e Ingeniería de Servicios (JCIS) Zotano et al. (2013)
- Los hallazgos preliminares sobre altos valores de α en trazas de servidor se publicaron en las 12th International Conference in Distributed Computing and Artificial Intelligence Zotano et al. (2015b)
- Los análisis iniciales sobre la presencia de α en el lado del cliente se presentó en Journal of Advances in Distributed Computing and Artificial Intelligence Gómez Zotano et al. (2015)
- Las consecuencias de estos nuevos valores en la configuración de cachés se publicó en el International Journal of Web Engineering and Technology Zotano et al. (2015a)

Además de las mencionadas, otras publicaciones abordan aplicaciones de lo tratado en la tesis en el entorno profesional de RTVE (ver apéndice A).

Tras este resumen, a continuación se detallan las principales contribuciones de la tesis, para luego mostrar líneas de trabajo futuro y aspectos que quedan por investigar. Esta sección va estar dividida en dos partes atendiendo a las cuestiones que se planteaban en la sección 1.2 de objetivos. Por un lado qué contribuciones se han hecho desde el punto de vista del análisis de la presencia de la función Zipf en el lado del servidor, y por otro lado en el cliente.

7.1. Contribuciones en la capa de servidor

Como se ha comentado anteriormente, el punto de partida de la tesis es la existencia de Zipfs en el lado de servidor con un valor de α acotado entre 0,6 y 1. Tras analizar 16 bancos de trazas actuales, se concluye que la función Zipf sigue modelando la frecuencia de solicitudes de objetos web. Esto permite

contestar afirmativamente a la primera cuestión que se planteó al iniciar el trabajo: si Zipf sigue modelando las frecuencias de acceso a objetos web en el lado de servidor con los volúmenes actuales de información y las tecnologías vigentes.

El hecho de que los 16 bancos correspondan a diferentes dominios sectoriales y que empleen diferentes tecnologías, permite contestar a la otra de las preguntas de la introducción y afirmar que las evidencias indican que se da una función Zipf en el lado del servidor independientemente de estos elementos. Además, Zipf aparecía también al considerar diferentes filtros sobre el banco de datos, como ventanas temporales o tipos de contenidos.

Este nuevo análisis sobre mayores y mejores bancos de datos ha identificado variaciones del α asociado a la Zipf, en línea con otra de las cuestiones que motivaban la tesis. En el capítulo 3 se ha encontrado que el valor de α es mucho mayor que antes, hasta un 87 % mayor que la cota que se indicaba en la bibliografía.

Sobre el impacto de programas automáticos como bots y crawlers, se ha encontrado que las peticiones que hacen también siguen una Zipf, si bien su *alpha* es un 300 % el valor indicado en la literatura para los usuarios.

Las consecuencias de este mayor valor de α también han sido analizadas en el capítulo 4, encontrando valores muy inferiores de recursos de caché para generar altas medidas de *hit-ratio*. Almacenando un 0,6 % del contenido total en caché, se consiguen valores de *hit-ratio* de un 84 %. Además de lo anterior, también se ha concluido que las políticas de caché con mejor rendimiento en el pasado siguen teniendo un rendimiento muy alto con estos nuevos valores de α.

Aparte del análisis de las cachés, otra contribución que se hace desde los trabajos de esta tesis es la capacidad de predecir el *hit ratio* de una muestra. En el capítulo 4 se ha hecho el desarrollo de las ecuaciones para contemplar todos los escenarios, con lo que la predictibilidad del *hit ratio* queda completada.

7.2. Contribuciones en la capa de cliente

La capa de cliente ha sido menos estudiada en la literatura sobre la función Zipf. Este trabajo de tesis planteó varias preguntas al respecto: si también se producía Zipf en este ámbito; en caso afirmativo cómo era su α; y si había dependencias entre el α obtenido analizando objetos web y el obtenido analizando páginas web.

La capa de cliente ha sido analizada en parte gracias a las contribuciones

de RTVE y en parte gracias a fuentes públicas, en concreto OJD (2016). Tras el análisis de estas fuentes hay una serie de tendencias no documentadas que se han encontrado. En concreto, en la capa de cliente, las aportaciones más relevantes han sido:

- En sitios como rtve.es, que tienen una Zipf en el servidor con $\alpha > 1$, el cliente también presenta un patrón de consumo Zipf con $\alpha > 1$, aunque con un α menor que en el servidor.

- Para los sitios web con un consumo masivo de contenido multimedia y de noticias (rtve.es, telecinco.es, canalsur.es, etc.) se puede ver que el comportamiento del usuario ha cambiado: de un usuario que permanece en un sitio web, navegando dentro del mismo utilizando los enlaces de las páginas, a un usuario que tiene un patrón de consumo más directo. Esto se denota por el aumento del número de visitas, el descenso del tiempo de visita y por el descenso de páginas vistas en una visita.

- Las sesiones de usuario en móvil duran menos e incluyen menos páginas que en otros dispositivos: el consumo del contenido es este caso es más compulsivo, y el papel del webmaster como recomendador y enlazador a otros contenidos no es efectivo.

- Por último, las redes sociales, la movilidad y la integración en los buscadores, se apuntan como explicación de este cambio de patrón de consumo por parte del usuario.

La relación entre las peticiones en la capa de cliente y la del servidor es quizá uno de los elementos que menos se ha estudiado en la bibliografía y donde más ha contribuido esta tesis. En la mayoría de los estudios se asume que basta estudiar el servidor para entender lo que está ocurriendo en el cliente. En esta tesis no se ha partido de dicha hipótesis, pero se ha querido buscar evidencias que la validen o refuten usando como guía el banco de datos obtenido de RTVE.

Se ha podido corroborar que los cambios en el patrón de consumo en el cliente afectan directamente al patrón que se obtiene en el servidor. Asi un incremento en el α del cliente implica un incremento en el α del servidor, aunque el incremento en el servidor es más pequeño. Se valida así el *trickle down effect* (Doyle et al., 2002). Esto podría cambiar en el futuro, ya que se ha encontrado que la técnica de rediseño de sitios web basada en redirecciones por user-agents provoca una redistribución de las frecuencias y hace que la distribución final en el servidor presente un α menor.

No obstante, también se ha podido comprobar que el servidor recogía una función Zipf con independencia del comportamiento en el cliente en los experimentos del capítulo 6. Los experimentos realizados sugieren que el comportamiento del usuario únicamente afecta al servidor cambiando la pendiente de la Zipf, pero no influye en si aparece o no. Esto es relevante porque invalida el estudio del comportamiento del usuario utilizando únicamente los logs del servidor, y porque sugiere que las Zipf se dan en servidor más por motivos estructurales del HTML (e.g. de cómo se construyen las páginas o de cómo se referencias los objetos) que por las acciones del usuario.

7.3. Trabajo futuro

Esta tesis ha perseguido ser exhaustiva en el análisis de todos los puntos de vista dentro del dominio de conocimiento. Por ello empezó por el servidor analizando evidencias y se llegó a un modelo que explica el comportamiento. Luego se buscaron las consecuencias en temas de caché. En ese punto hay un punto no resuelto que tiene que ver con las técnicas de caché a utilizar si se asume que el contenido caduca. En el capítulo 4 se analizó la mejor política de caché siguiendo benchmarks identificados en la literatura, pero no se tuvo en cuenta la caducidad de los objetos. Esto podría ser de interés, ya que podría haber mecanismos de consistencia de datos que aprovechen el patrón de las distribuciones Zipf con $\alpha > 1$ no estudiados.

También en línea con lo anterior, hay una fuerte tendencia a distribuir los contenidos usando CDNs (redes de cacheo de contenidos). Teniendo en cuenta lo que ha descendido el coste de servicios en la nube, sería de interés validar si el modelo de cacheo podría ser sustituido por el modelo de distribución de carga (Cloud), máxime cuando mucho del contenido que se genera es dinámico. El contenido estático se beneficia del cacheo distribuido ya que no implica carga, pero en el dinámico quizá sí que haya otras opciones, sobre todo con las posibilidades que ofrece la computación en la nube.

Respecto al lado del cliente, sería interesante validar si los factores que se han determinado como causantes del consumo compulsivo lo son en otros dominios, más allá de rtve.es. Estudios adicionales podrían determinar qué factores son los que hacen que el usuario movilizado sea tan compulsivo. Quizá dicho estudio transcienda el tema técnico y tenga más que ver con el sociológico, pero es de interés entender el por qué se consume con esta intensidad.

Finalmente, una línea que no se ha investigado es el indagar en por qué se dan Zipfs en el servidor. En esta tesis se ha ahondado en el impacto del comportamiento del usuario en la distribución Zipf encontrada en el servidor,

y se ha podido comprobar que hay casos en los que las acciones del usuario no son la causa de dicho comportamiento. Sería relevante ahondar en este asunto para zanjar la cuestión.

Apéndice A

Publicaciones de la tesis

En este apéndice se muestra el listado de publicaciones y ponencias que se han realizado durante los trabajos de investigación. Se distinguen entre publicaciones en revista, congreso y ponencias en foros industriales. Estas últimas se presentan de forma independiente al resto por no generarse tras un proceso de revisión por pares y se incluyen como evidencia de la conexión del trabajo realizado con el ámbito industrial.

A.1. Ponencias en congreso

A.1.1. Diseño de una caché de objetos para servicios del tipo RTVE a la carta

A.1.1.1. Resumen

Los servicios del tipo RTVE alacarta que se construyen sobre capas con lógica de negocio basadas en objetos, suelen presentar problemas de rendimiento por el número de peticiones que tienen que soportar y por la naturaleza del modelo de negocio. El hecho diferencial de estos servicios es que la información se actualiza cada muy poco tiempo y que los cambios tienen que ser accesibles de forma casi inmediata a futuras peticiones. Para solventar los problemas de rendimiento se plantea el uso de caches orientadas a objetos de negocios que cumplan con los requisitos de inmediatez de la información, pero permitiendo ofrecer el servicio con garantías.

A.1.1.2. Palabras clave

cache, servicios, objetos, domain driven design.

A.1.1.3. Referencia

Zotano, M. G., Sanz, J. G., & Pavón, J. (2013). Diseño de una caché de objetos para servicios del tipo RTVE a la carta. Actas de las IX Jornadas de Ciencia e Ingeniería de Servicios (JCIS) en Madrid. Páginas 67-79.

A.1.2. Analysis of Web Objects Distribution

A.1.2.1. Resumen

Understanding how the web objects of a website are demanded is relevant for the design and implementation of techniques that assure a good quality of service. Several authors have studied generic profiles for web access, concluding that they resemble a Zipf distribution, but further evidences were missing. This paper contributes with additional empirical evidences that confirm that a Zipf distribution is present in different domains and that its form has changed from past studies. More specifically, the α parameter has become higher than one, as a consequence that the popularity factor has become more critical than before. This analysis also considers the impact of web technologies on the characterization of web traffic.

A.1.2.2. Palabras clave

Web traffic analysis Web technology Zipf distribution Cache policy.

A.1.2.3. Referencia

Zotano, M. G., Sanz, J. G., & Pavón, J. (2015). Analysis of Web Objects Distribution. En: Distributed Computing and Artificial Intelligence, 12th International Conference. Advances in Intelligent Systems and Computing 373, pp. 105-112. Springer.

A.2. Artículos en revista

A.2.1. Impact of traffic distribution on web cache performance

A.2.1.1. Resumen

Caches are a critical element of web-based information systems. Understanding the expected behaviour of cache policies is especially important for

achieving good quality of service. Existing works have suggested that the behaviour of the web demand can be modelled as a Zipf distribution with $\alpha \leq 1$. New evidence, which is presented in this paper, shows that today websites are following Zipf distributions with $\alpha > 1$. This article analyses real logs obtained from the client layer of high traffic websites. The main result of this article is that under these conditions, the cache hit ratio can be extremely high with a very small cache size. This means that a very expensive and high resource demanding cache is not needed for effective implementation: a cache size equal to 0.6 % of the working set is enough to reach more than 80 % of hit ratio, once the right replacement policy has been chosen.

A.2.1.2. Palabras clave

Website traffic distribution, web cache performance, Zipf distribution, web logs, cache hit rate, web performance, web information systems, quality of service, QoS, cache size, replacement policy.

A.2.1.3. Referencia

Zotano, M. G., Gómez-Sanz, J., & Pavón, J. (2015). Impact of traffic distribution on web cache Performance. International Journal of Web Engineering and Technology, 10(3), 202-213.

A.2.2. User Behavior in Mass Media Website

A.2.2.1. Resumen

Mass media websites can be worthy to understand user trends in web services. RTVE, the National Broadcaster in Spain is a sample of such kind of service. The analysis of trends points to a shorter user interaction over the last three years, and a more straight access to content. Besides the number of pages consumed in a visit is becoming smaller as well. This article reviews these trends with data obtained from public sources, and analyze the distribution of web pages in the client layer and the corresponding distribution observed in the server layer. These two distributions can be characterized by Zipf-like distributions and α, the degree of disparity in the popularity distribution, is calculated for both. In all cases α is higher to one implying a huge concentration of popularity on a few objects.

A.2.2.2. Palabras clave

Web traffic analysis; Zipf distribution, Web user patterns, Mass Media Website.

A.2.2.3. Referencia

Zotano, M. G., Gómez-Sanz, J., & Pavón, J. (2015). User Behavior in Mass Media Website. Advances in Distributed Computing and Artificial Intelligence Journal, 4 (3), 47-56.

A.3. Ponencias en foros industriales

Estas ponencias son trabajos realizados dentro de la actividad profesional del doctorando que se han presentado en foros industriales. Parte de los contenidos presentados guardan relación con este trabajo de tesis. A continuación se describe cada uno y su vínculo con este trabajo.

A.3.1. Distribución de multimedia en aplicaciones móviles a través de CDNs.

A.3.1.1. Resumen

En esta ponencia se describen los mecanismos que emplea RTVE para la distribución de contenidos en directo y bajo demanda. Además se justifica el uso de CDN, y se analiza los patrones de consumo de vídeos por parte de los usuarios. La ponencia trató sobre cómo distribuye RTVE los contenidos a través de CDNs, y cómo funcionan los distintos modelos de caché, dependiendo del tipo de consumo multimedia. Parte de la información forma parte del capítulo 2, del estado del arte sobre cómo funcionan las cachés. También hay contribuciones del capítulo 3 referente al capítulo de evidencias de Zipf en el servidor para el caso de RTVE, específicamente, el análisis de tipos de objetos web, atendiendo a su frecuencia de acceso en servidores.

A.3.1.2. Palabras clave

CDN, patrones de consumo, distribución multimedia

A.3.1.3. Referencia

Zotano, M.G. (2013). Distribución de multimedia en aplicaciones móviles a través de CDNs. Codemotion 2013. Madrid.

A.3.2. Development pipeline with Node.js

A.3.2.1. Resumen

En esta ponencia se describen el modelo de desarrollo y el de despliegue para NodeJS que se emplean en RTVE. Además, se analizan los mecanismos y políticas de caché que se emplean para garantizar la escalabilidad de los servicios así desarrollados. En la ponencia se trata el modelo de despliegue de aplicaciones de RTVE, así como la forma en que se explotan algunos de los resultados obtenidos en los capítulos 3 y 4. Concretamente se hace énfasis en los modelos de coherencia y consistencia de cachés empleados en RTVE.

A.3.2.2. Palabras clave

NodeJS, políticas de caché, gestión de dependencias, CDN, patrones de consumo

A.3.2.3. Referencia

Zotano, M.G. (2015). Development pipeline with NodeJS. EBU DEVCON 2015. Ginebra.

A.3.3. Use case for Olympic Games

A.3.3.1. Resumen

En la ponencia se describen los servicios que RTVE desarrolló para la cobertura de los Juegos Olímpicos de Rio 2016. Se hace especial hincapié en las tecnologías y técnicas que garantizan la escalabilidad de las soluciones, principalmente, CDNs y modelos de cacheo de contenidos. En esta ponencia se analiza técnicamente el despliegue para el evento de Juegos Olímpicos, y destaca los modelos de escalabilidad que se montaron: estos utilizan los resultados de los capítulos 3 y 4 de la tesis para optimizar el cacheo en servidor, y predecir el nivel de cacheo en cliente. Cabe destacar el modelo de coherencia y consistencia de cachés empleados, basados en el push de objetos, así como la diferenciación de políticas basadas en el tipo de contenido.

A.3.3.2. Palabras clave

CDN, juegos olímpicos, modelos de caché, GDSF, escalabilidad, streaming

A.3.3.3. Referencia

Zotano, M.G. (2016). Use case for Olympic Games. EBU Broadthinking 2016. Ginebra.

Bibliografía

Almeida, V., Bestavros, A., Crovella, M. y De Oliveira, A. Characterizing reference locality in the www. En *Parallel and Distributed Information Systems, 1996., Fourth International Conference on*, páginas 92–103. IEEE, 1996.

Almeida, V., Cesario, M., Fonseca, R., Meira Jr, W. y Murta, C. Analyzing the behavior of a proxy server in the light of regional and cultural issues. En *Proceedings of WCW*. 1998.

Apache, F. Apache http server version 1.3. 2004. [Online; http://httpd.apache.org/docs/1.3/logs.html; accedido 15-Juliol-2015].

Arlitt, M., Cherkasova, L., Dilley, J., Friedrich, R. y Jin, T. Evaluating content management techniques for web proxy caches. *ACM SIGMETRICS Performance Evaluation Review*, vol. 27(4), páginas 3–11, 2000.

Arlitt, M. y Jin, T. A workload characterization study of the 1998 world cup web site. *Network, IEEE*, vol. 14(3), páginas 30–37, 2000.

Bowler, M. Htmlunit. 2002.

Breslau, L., Cao, P., Fan, L., Phillips, G. y Shenker, S. Web caching and zipf-like distributions: Evidence and implications. En *INFOCOM'99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 1, páginas 126–134. IEEE, 1999.

Busari, M. y Williamson, C. On the sensitivity of web proxy cache performance to workload characteristics. En *INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 3, páginas 1225–1234. IEEE, 2001.

Calzarossa, M. C., Massari, L. y Tessera, D. Workload characterization: A survey revisited. *ACM Computing Surveys (CSUR)*, vol. 48(3), página 48, 2016.

CHALLENGER, J. R., DANTZIG, P., IYENGAR, A., SQUILLANTE, M. S. y ZHANG, L. Efficiently serving dynamic data at highly accessed web sites. *Networking, IEEE/ACM Transactions on*, vol. 12(2), páginas 233–246, 2004.

CHERKASOVA, L. y CIARDO, G. Role of aging, frequency, and size in web cache replacement policies. En *International Conference on High-Performance Computing and Networking*, páginas 114–123. Springer, 2001.

CHESIRE, M., WOLMAN, A., VOELKER, G. M. y LEVY, H. M. Measurement and analysis of a streaming media workload. En *USITS*, vol. 1, páginas 1–1. 2001.

CLAUSET, A., SHALIZI, C. R. y NEWMAN, M. E. Power-law distributions in empirical data. *SIAM review*, vol. 51(4), páginas 661–703, 2009.

CLAUSET, A., SHALIZI, C. R. y NEWMAN, M. E. Implementations for fitting zipf. 2013.

CLOUD, A. M. Metric descriptions. 2016. [Online; https://marketing.adobe.com; accedido 20-April-2016].

CUNHA, C. R., BESTAVROS, A. y CROVELLA, M. E. Characteristics of www client-based traces. Informe técnico, Boston University Computer Science Department, 1995.

DOYLE, R. P., CHASE, J. S., GADDE, S. y VAHDAT, A. M. The trickle-down effect: Web caching and server request distribution. *Computer Communications*, vol. 25(4), páginas 345–356, 2002.

FIELDING, R., GETTYS, J., MOGUL, J., FRYSTYK, H., MASINTER, L., LEACH, P. y BERNERS-LEE, T. Hypertext transfer protocol–http/1.1. Informe técnico, 1999.

FOG HEEN, T., LYNGSTOL, K. y RENARD, J. Ehcache 2.7 documentation. 2013. [Online; https://www.varnish-software.com/system/files/web_downloads/varnish-book.pdf; accedido 2-Enero-2015].

FREED, N. y BORENSTEIN, N. Multipurpose internet mail extensions (mime) part one: Format of internet message bodies. Informe técnico, 1996.

GILL, P., ARLITT, M., LI, Z. y MAHANTI, A. Youtube traffic characterization: a view from the edge. En *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, páginas 15–28. ACM, 2007.

GILLESPIE, C. S. Fitting heavy tailed distributions: the powerlaw package. *arXiv preprint arXiv:1407.3492*, 2014.

GLASSMAN, S. A caching relay for the world wide web. *Computer Networks and ISDN Systems*, vol. 27(2), páginas 165–173, 1994.

GÓMEZ ZOTANO, M., GOMÉZ-SANZ, J. y PAVÓN, J. User behavior in mass media websites. *Advances in Distributed Computing and Artificial Intelligence Journal*, vol. 4(3), páginas 47–56, 2015.

GONZALEZ-CANETE, F., CASILARI, E. y TRIVINO-CABRERA, A. Characterizing document types to evaluate web cache replacement policies. En *Universal Multiservice Networks, 2007. ECUMN'07. Fourth European Conference on*, páginas 3–11. IEEE, 2007.

GOOGLE. Google analytics web site. 2016. [Online; https://www.google.com/analytics/;accedido 17-April-2016].

GRACE, L., MAHESWARI, V. y NAGAMALAI, D. Analysis of web logs and web user in web mining. *arXiv preprint arXiv:1101.5668*, 2011.

HOOPER, A. Green computing. *Communication of the ACM*, vol. 51(10), páginas 11–13, 2008.

HUANG, Q., BIRMAN, K., VAN RENESSE, R., LLOYD, W., KUMAR, S. y LI, H. C. An analysis of facebook photo caching. En *Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles*, páginas 167–181. ACM, 2013.

KRASHAKOV, S. A., TESLYUK, A. B. y SHCHUR, L. N. On the universality of rank distributions of website popularity. *Computer Networks*, vol. 50(11), páginas 1769–1780, 2006.

LUKE, S., CIOFFI-REVILLA, C., PANAIT, L., SULLIVAN, K. y BALAN, G. Mason: A multiagent simulation environment. *Simulation*, vol. 81(7), páginas 517–527, 2005.

MAHANTI, A., CARLSSON, N., ARLITT, M. y WILLIAMSON, C. A tale of the tails: Power-laws in internet measurements. *Network, IEEE*, vol. 27(1), páginas 59–64, 2013.

MAHANTI, A., WILLIAMSON, C. y EAGER, D. Traffic analysis of a web proxy caching hierarchy. *Network, IEEE*, vol. 14(3), páginas 16–23, 2000.

MAHANTI, A., WILLIAMSON, C. y WU, L. Workload characterization of a large systems conference web server. En *Communication Networks and Services Research Conference, 2009. CNSR'09. Seventh Annual*, páginas 55–64. IEEE, 2009.

MITZENMACHER, M. A brief history of generative models for power law and lognormal distributions. *Internet mathematics*, vol. 1(2), páginas 226–251, 2004.

NAIR, T. y JAYAREKHA, P. A rank based replacement policy for multimedia server cache using zipf-like law. *arXiv preprint arXiv:1003.4062*, 2010.

NISHIKAWA, N., HOSOKAWA, T., MORI, Y., YOSHIDA, K. y TSUJI, H. Memory-based architecture for distributed www caching proxy. *Computer Networks and ISDN Systems*, vol. 30(1), páginas 205–214, 1998.

OJD. Evolucion audiencias rtve.es. 2016. [Online; http://www.ojdinteractiva.es/medios-digitales; accedido 17-Abril-2016].

PITA FERNÁNDEZ, S. y PÉRTEGA DÍAZ, S. Relación entre variables cuantitativas. *Cad Aten Primaria*, vol. 4, páginas 141–4, 1997.

PODLIPNIG, S. y BÖSZÖRMENYI, L. A survey of web cache replacement strategies. *ACM Computing Surveys (CSUR)*, vol. 35(4), páginas 374–398, 2003.

ROADKNIGHT, C., MARSHALL, I. y VEARER, D. File popularity characterisation. *ACM Sigmetrics Performance Evaluation Review*, vol. 27(4), páginas 45–50, 2000.

ROMANO, S. y ELAARAG, H. A quantitative study of web cache replacement strategies using simulation. *Simulation*, página 0037549711414152, 2011.

SHI, L., GU, Z., WEI, L. y SHI, Y. Quantitative analysis of zipf's law on web cache. En *Parallel and Distributed Processing and Applications*, páginas 845–852. Springer, 2005.

SHI, L., GU, Z., WEI, L. y SHI, Y. An applicative study of zipf's law on web cache. *International Journal of Information Technology*, vol. 12(4), páginas 49–58, 2006.

SQUID, E. Squid documentation. 2015. [Online; http://www.squid-cache.org/Doc/; accedido 12-Agosto-2015].

TEAM, T. Ehcache 2.7 documentation. 2013. [Online; http://ehcache. org/documentation; accedido 2-Enero-2015].

URDANETA, G., PIERRE, G. y VAN STEEN, M. Wikipedia workload analysis for decentralized hosting. *Computer Networks*, vol. 53(11), páginas 1830–1845, 2009.

WONG, K.-Y. Web cache replacement policies: a pragmatic approach. *IEEE Network*, vol. 20(1), páginas 28–34, 2006.

ZHOU, X., WEI, J. y XU, C.-Z. Quality-of-service differentiation on the internet: A taxonomy. *Journal of Network and Computer Applications*, vol. 30(1), páginas 354–383, 2007.

ZIPF, G. K. *Human Behaviour and the Principle of Least-Effort*. Addison-Wesley, Cambridge MA, 1949.

ZOTANO, M. G., GÓMEZ-SANZ, J. y PAVÓN, J. Impact of traffic distribution on web cache performance. *International Journal of Web Engineering and Technology*, vol. 10(3), páginas 202–213, 2015a.

ZOTANO, M. G., SANZ, J. G. y PAVÓN, J. Diseño de una caché de objetos para servicios del tipo rtve a la carta. *Actas de las IX Jornadas de Ciencia e Ingeniería de Servicios (JCIS) en Madrid*, páginas 67–79, 2013.

ZOTANO, M. G., SANZ, J. G. y PAVÓN, J. Analysis of web objects distribution. En *Distributed Computing and Artificial Intelligence, 12th International Conference*, páginas 105–112. Springer, 2015b.